PERANCANGAN DAN IMPLEMENTASI PERANGKAT LUNAK UNTUK PENGELOMPOKKAN DAERAH PRODUKSI MINYAK BUMI DENGAN METODE KOHONEN

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Pada
Jurusan Teknik Informatika

oleh :

FITRI INSANI
10551001460

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU
2010

PERANCANGAN DAN IMPLEMENTASI PERANGKAT LUNAK UNTUK PENGELOMPOKKAN DAERAH PRODUKSI MINYAK BUMI DENGAN METODE KOHONEN

FITRI INSANI

10551001460

Tanggal Sidang : 11 Juni 2010

Periode Wisuda : 15 Juli 2010

Jurusan Teknik Informatika

Fakultas Sains dan Teknologi

Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Teknologi telah berkembang dengan pesat seiring dengan berjalannya waktu. Hal ini ditandai dengan munculnya berbagai macam komputerisasi disegala bidang. Salah satu kemampuan yang dapat dilakukan adalah memanfaatkan teknologi tersebut untuk melakukan pengelompokan data (*cluster*). Perusahaan perminyakan memiliki banyak sumur minyak sehingga memerlukan dilakukannya pengelompokkan (*clustering*) sumur minyak.

Aplikasi ini menggunakan algoritma Self Organizing Maps (SOM) untuk melakukan analisa cluster dan Java 3D sebagai bahasa pemrograman. Data sekunder sumur minyak digunakan untuk keperluan proses clusterisasi. Data sumur minyak dipersiapkan dengan melalui proses transformasi ke bentuk yang dapat diolah oleh algoritma SOM menggunakan *min max* normalisasi. Kemudian data diolah menggunakan algoritma SOM untuk menghasilkan *cluster-cluster* data. Hasil *cluster-cluster* ini ditampilkan dalam bentuk *bubblechart* 3D.

Dengan menggunakan output dari aplikasi ini, yang berupa *cluster* daerah produksi sumur minyak, pengambil keputusan dapat dengan mudah menganalisa tiap *cluster*. Informasi ini dapat membantu pengguna dalam pengambilan keputusan.

Kata Kunci : *Cluster Analysis*, Daerah Produksi Sumur Minyak, *Self Organizing Maps* (SOM)

SOFTWARE DESIGN AND IMPLEMENTATION FOR CLUSTERING WELL PRODUCTION BY KOHONEN'S METHOD

FITRI INSANI

10551001460

Date of Final Exam : June 11th, 2010

Graduation Ceremony Period : July 15th, 2010

Informatics Departement

Faculty of Sciences and Technology

State Islamic University of Sultan Syarif Kasim Riau

ABSTRACT

Technology has grown rapidly as the time goes on. This is signed with many computerization in all area. One of ability which can be done is utilize that technology to do data agglomeration (cluster). Oil company which have a lot of well, requires to be done its clustering process.

This application uses Self Organizing Maps algorithm for Cluster Analysis and Java 3D as programming language. The sekunder well production data is used for clustering process. The well production data is prepared through transforming into a form that can be processed by SOM Algorithm used min max normalization. After that, the data is clustered with the SOM algorithm. The result of that clusters is displayed into a form bubblechart 3D.

By using the output / result of this application, that are the clusters of well production, the user can analyse the statistics of each cluster. This information can give a support for user to make a decision.

***Key Words**: Cluster Analysis, Self Organizing Maps (SOM), Well Production*

DAFTAR ISI

Halaman

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sekitar tahun 1950-an, komputer masih merupakan barang langka di dunia (Nugroho, 1993). Dari asal kata "*to compute*" computer bararti alat penghitung (Daryanto, 2003). Namun seiring dengan perkembangan zaman, maka peran komputer semakin mendominasi kehidupan. Lebih dari itu, komputer diharapkan dapat digunakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia baik dalam bidang pendidikan, kesehatan, industri, dan kehidupan sehari-hari sehingga peran komputer dan manusia akan saling melengkapi. (Kusumadewi, 2003)

Semakin berkembangnya teknologi komputasi menjadikan sebuah komputer memiliki kemampuan yang tinggi dalam membantu pekarjaan manusia. Salah satu kemampuan yang dapat dilakukan adalah memanfaatkan teknologi tersebut untuk melakukan pengelompokan data (*cluster*). Perusahaan perminyakan merupakan salah satu perusahaan yang membutuhkan pengelompokkan data.

Perusahaan perminyakan yang memiliki banyak sumur minyak memerlukan dilakukannya pengelompokkan (*clustering*) sumur minyak. Pada dasarnya pengelompokkan sumur minyak itu dilakukan untuk melihat penyebaran minyak dilihat dari tingkat produksi, *water cut, pressure,* dan lain sebagainya dengan tujuan untuk mengetahui tingkat produksi suatu sumur minyak.

Jika *water cut* lebih dari 90% atau 100% dimana sebagian besar hanya air yang didapat dari suatu sumur minyak maka produksi minyak tersebut dapat dihentikan, tetapi hal tersebut tergantung dari keputusan pihak perusahaan minyak.

Tindakan yang dilakukan untuk menaikkan laju produksi dari suatu sumur minyak (stimulasi) dengan tiga macam cara yaitu pertama *wellbore cleanup* dimana fluida *treatment* dipompakan hanya ke dalam sumur, tidak sampai ke formasi. Kedua stimulasi matrik, fluida diinjeksikan ke dalam formasi hidrokarbon tanpa memecahkannya. Ketiga teknik *fracturing*, fluida diinjeksikan ke dalam formasi dengan laju dan tekanan tertentu sehingga formasi akan pecah atau merekah. (http://wiki.migas-indonesia.net/index.php/Minyak_Bumi).

Dengan adanya pengelompokkan sumur minyak, perusahaan perminyakan dapat mengetahui tingkat produksi suatu sumur minyak dan membantu pihak perusahaan mengambil kebijakan untuk menghentikan atau tetap melanjutkan operasional produksi suatu sumur minyak .

Pada Tugas Akhir ini akan dikembangkan suatu perangkat lunak untuk melakukan pengelompokkan daerah produksi sumur minyak bumi menggunakan Jaringan Saraf Tiruan (JST) dengan metode *Kohonen*. Masukan (*input*) pada JST berupa data produksi minyak (*oil production*) dan produksi air (*waterprod*). Keluaran berupa pengelompokkan sumur minyak.

Metode yang digunakan dalam tugas akhir ini adalah jaringan syaraf tiruan kohonen atau *Self Organizing Maps* (*SOM*). Pada metode ini, suatu lapisan yang berisi neuron-neuron akan menyusun dirinya sendiri berdasarkan input nilai

tertentu dalam suatu kelompok yang dikenal dengan istilah *cluster* (Kesumadewi, 2003). Dengan memanfaatkan metode *kohonen* atau *Self Organizing Maps* (*SOM*) yang merupakan salah satu metode dari jaringan syaraf tiruan ini, diharapkan sistem melakukan pengelompokkan sumur minyak bumi dengan baik.

Jaringan Saraf Tiruan telah diterapkan untuk menyelesaikan masalah pengelompokkan pada beberapa bidang seperti *Clustering Specimen* Daun Dikotiledon (Madarum, 2003) dan *Multilevel Learning in Kohonen Som Network for Classification Problems* (Yusof, 2006).

Tugas akhir ini merupakan Tugas Akhir yang dikembangkan dari tugas akhir Sumini Iriani (2007). Dalam Tugas Akhir tersebut telah dirancang sebuah sistem yaitu sistem yang bisa mengelompokkan daerah produksi minyak bumi dengan metode *K-Means* .

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan diatas dapat diambil suatu rumusan masalah yang akan dibahas yaitu: bagaimana merancang dan membangun suatu aplikasi jaringan syaraf tiruan untuk melakukan pengelompokkan daerah produksi minyak bumi menggunakan metode *Kohonen*.

1.3 Batasan Masalah

Agar pembahasan sesuai dengan tujuan penulisan, maka ruang lingkup batasan masalah yang disajikan adalah sebagai berikut:

1. *Input* yang dimasukkan berupa data sumur produksi (*well roduction*) yaitu produksi minyak (*oil production*), produksi air (*water prod*) dan produksi gas (*gas production*).
2. *Output* berupa visualisasi *bubble chart* 3 dimensi.
3. Jumlah cluster sebanyak 3 *cluster,* yaitu penuh, kering dan separuh penuh.

1.4 Tujuan

Adapun tujuan dalam tugas akhir ini adalah menganalisis, merancang dan mengimplementasikan sebuah aplikasi pengelompokkan daerah produksi minyak bumi dengan menggunakan jaringan syaraf tiruan *Kohonen*.

1.5 Sistematika Penulisan

Laporan tugas akhir ini terdiri dari enam bab, dengan sistematika penulisan sebagai berikut:

BAB I Pendahuluan

Bab ini berisi tentang deskripsi umum dari Tugas Akhir ini, yang meliputi latar belakang permasalahan, rumusan masalah, batasan masalah, tujuan dari pembahasan dan sistematika penulisan Tugas Akhir.

BAB II Landasan Teori

Bab ini akan membahas teori-teori yang berhubungan dengan pembahasan tugas akhir ini. Teori yang diangkat yaitu mengenai teori-teori tentang karakteristik jaringan syaraf biologis, karakteristik

jaringan syaraf tiruan, proses pembelajaran jaringan syaraf tiruan, algoritma jaringan syaraf tiruan dengan metode *Kohonen*, dan pembahasan mengenai minyak bumi.

BAB III Metodologi Penelitian

Bab ini akan membahas tentang metodologi penelitian yang digunakan dalam penyusunan tugas akhir ini.

BAB IV Analisis dan Perancangan

Berisikan tentang analisis pembahasan mengenai metode *Kohonen* yang diterapkan dengan menggunakan jaringan syaraf tiruan. Dan dibuat suatu rancangan aplikasi pengelompokkan daerah produksi minyak bumi dengan menerapkan jaringan syaraf tiruan *Kohonen.*

BAB V Implementasi dan Pengujian

Bab ini berisi penjelasan mengenai batasan implementasi, lingkungan implementasi dan hasil dari implementasi, serta menjelaskan pengujian perangkat lunak dan hasil pengujian.

BAB VI Penutup

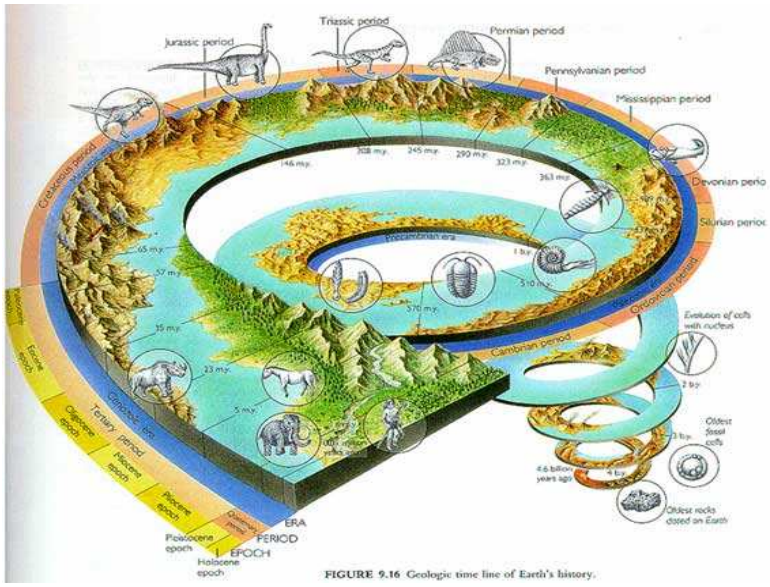
Bab ini berisikan kesimpulan dari Tugas Akhir yang dibuat dan menjelaskan saran-saran penulis kepada pembaca agar aplikasi jaringan syaraf tiruan untuk pengenalan karakter yang dibuat ini dapat dikembangkan lagi.

BAB II

LANDASAN TEORI

2.1 Minyak Bumi

Minyak bumi adalah cairan kental, coklat gelap, atau kehijauan yang mudah terbakar, yang berada di lapisan atas dari beberapa area di kerak bumi. Minyak bumi terdiri dari campuran kompleks dari berbagai hidrokarbon. (http://id.wikipedia.org/wiki/Minyak_bumi)



Gambar 2.1. Skala Waktu Geologi (http://wiki.migas-indonesia.net/index.php/Minyak_Bumi)

Geologi minyak bumi adalah salah satu cabang ilmu geologi untuk mengetahui keberadaan minyak bumi di bawah tanah, kemudian mengeksplorasi dan memproduksinya. Secara umum ada dua jenis geologi minyak bumi, yaitu

geologi eksplorasi minyak bumi yang mencakup pencarian minyak bumi dan geologi produksi minyak bumi. Produksi minyak bumi dalam bidang perminyakan bukan diartikan untuk membuat minyak bumi, tetapi hanyalah membuat fasilitas untuk mengalirkan minyak bumi dari bawah tanah ke atas permukaan tanah, dengan menggunakan pemboran dan pompa-pompa. (http://id.wikipedia.org/wiki/ Geologi_minyak_bumi)

Teori keberadaan minyak bumi ada dua buah, yaitu teori organik dan teori anorganik. Teori organik sekarang ini banyak dianut oleh para ahli geologi, dimana minyak bumi dipercayai dihasilkan oleh sisa-sisa organisma yang sudah mati berjuta-juta tahun yang lalu. Sedangkan teori anorganik kebanyakan berkembang di Eropa Timur dan Rusia di mana para ahli mempercayai bahwa minyak bumi dapat dihasilkan bukan dari bahan organik. (http://id.wikipedia.org/wiki/Geologi_minyak_bumi)

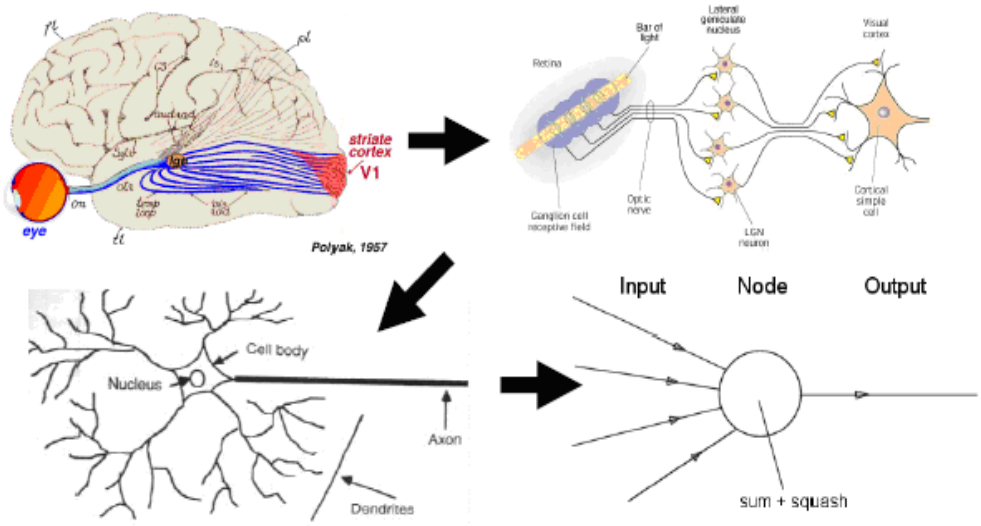
2.1.1 Eksplorasi Minyak Bumi

Eksplorasi atau pencarian minyak bumi merupakan suatu kajian panjang yang melibatkan beberapa bidang kajian kebumian dan ilmu eksak. Untuk kajian dasar, riset dilakukan oleh para geologis, yaitu orang-orang yang menguasai ilmu kebumian. Mereka adalah orang yang bertanggung jawab atas pencarian hidrokarbon tersebut. Minyak di dalam bumi bukan berupa wadah yang menyerupai danau, tetapi berada di dalam pori-pori batuan bercampur bersama air. (http://id.wikipedia.org/wiki/Eksplorasi_minyak_bumi

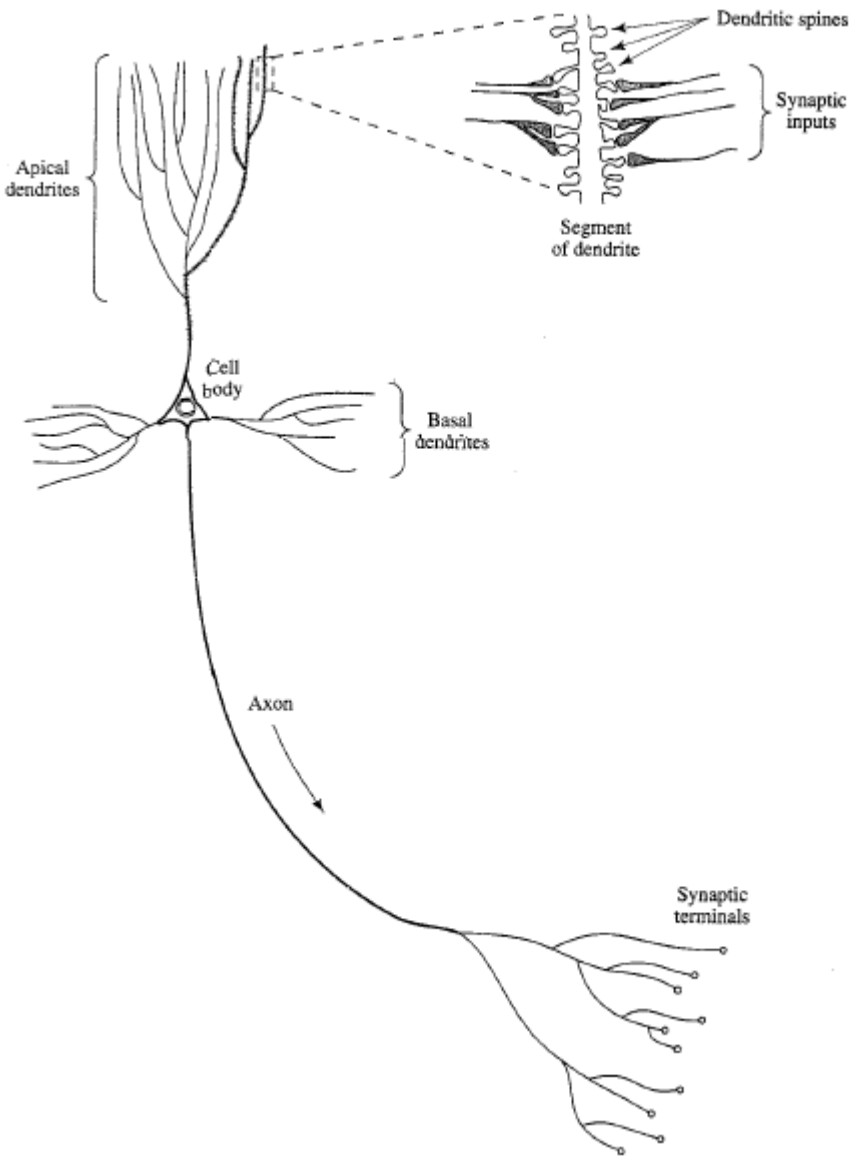
2.2 Jaringan Saraf Manusia

Otak manusia diperkirakan terdiri atas 10^{11} sel saraf (*neuron*). Di otak inilah terdapat fungsi-fungsi yang sangat banyak dan rumit, diantaranya adalah ingatan, belajar, bahasa, dan lain-lain. Untuk membentuk fungsi-fungsi itu setiap sel saraf akan saling berhubungan membentuk jaringan yang sangat rumit yang disebut jaringan saraf. (Hermawan, 2006)

Setiap neuron mempunyai kemampuan untuk menerima, memproses, dan menghantarkan sinyal elektro kimiawi melalui jalur-jalur saraf.



Gambar 2.2. Pemodelan Jaringan Saraf Tiruan (http://student.eepis-its.edu/~prara/Semester%205/AI/Bab%208%20Jaringan%20Syaraf%20Tiruan.pdf)



Gambar 2. 3. Struktur Sebuah Neuron (Haykin, 2005)

2.3 Jaringan Saraf Tiruan

Pada tahun 1943 oleh seorang ahli saraf Warren Mc Culloch dan seorang ahli logika Walter Pits merancang model formal yang pertama kali sebagai perhitungan dasar *neuron*. (Hermawan, 2006)

2.3.1 Definisi Jaringan Saraf Tiruan

Jaringan Saraf Tiruan adalah sebuah kelompok pengolahan elemen dalam suatu kelompok yang khusus membuat perhitungan sendiri dan memberikan hasilnya kepada kelompok kedua atau berikutnya. Setiap sub-kelompok menurut gilirannya harus membuat perhitungan sendiri dan memberikan hasilnya untuk *subgroup* atau kelompok yang belum melakukan perhitungan. Pada akhirnya sebuah kelompok dari satu atau beberapa pengolahan elemen tersebut menghasilkan keluaran (*output*) dari jaringan. (Rao dan Rao, 1993)

Jaringan saraf tiruan (JST) (*artificial neural network (ANN),* atau juga disebut *simulated neural network (SNN),* atau umumnya hanya disebut *neural network (NN)*), adalah sekelompok jaringan saraf *(neuron)* buatan yang menggunakan model matematis atau komputasi untuk pemrosesan informasi berdasarkan pendekatan terhubung pada komputasi. Pada kebanyakan kasus, JST merupakan sistem adaptif yang merubah strukturnya berdasarkan informasi eksternal maupun internal yang mengalir melalui jaringan tersebut. Istilah yang lebih praktis untuk jaringan syaraf adalah bahwa dia merupakan alat pemodelan data statistik non-linier. JST dapat digunakan untuk memodelkan hubungan yang kompleks antara *input* dan *output* untuk menemukan pola-pola data. (http://id.wikipedia.org/wiki/Jaringan_saraf_tiruan).

2.3.2 Aplikasi Jaringan Saraf Tiruan

Aplikasi jaringan saraf tiruan yang digunakan antara lain di bidang :

1. Otomotif, Sistem pembelajaran mobil otomatis

2. Sistem penerbangan udara, autopilot pesawat terbang dengan kemampuan maksimal, simulasi jalur penerbangan.
3. Sistem Pertahanan, system pelacakan, deteksi suatu objek, pengenalan *interface*, radar dan isyarat gambar.
4. Suara, pengenalan suara, penggolongan huruf hidup, pemindahan bentuk teks ke suara.
5. Telekomunikasi, kompresi data dan gambar.
6. Perdagangan, analisis pasar, system pendukung perdagangan bursa.
7. Kesehatan, analisis sel penyakit kanker, optimalisasi waktu pencangkokan.

2.3.3 Kelebihan Jaringan Saraf Tiruan

Kelebihan Jaringan Saraf Tiruan (Hermawan, 2006):

1. Kemampuan mengakuisisi pengetahuan walaupun dalam kondisi ada gangguan dan ketidakpastian.
2. Kemampuan mempresentasikan pengetahuan secara fleksibel. Jaringan saraf tiruan dapat menciptakan sendiri representasi melalui pengetahuan diri sendiri atau kemampuan belajar (*self organizing*)
3. Kemampuan untuk memberikan toleransi atas suatu distorsi (*error* atau *fault*), dimana gangguan kecil pada data dapat dianggap hanya sebagai *noise*(guncangan) belaka.
4. Kemampuan memproses pengetahuan secara efesien karena memakai sistem parallel, sehingga waktu yang diperlukan untuk mengoperasikannya menjadi lebih singkat.

2.4 Agoritma Belajar pada Jaringan Saraf Tiruan

Ada tiga jenis algoritma belajar, yaitu : (Jha, 2007)

1. *Supervised Learning* (proses belajar terawasi), pada pembelajaran ini target tersedia (diberikan oleh penyelia/ guru). Contohnya antara lain *Delta Rule* (Widrow dan Hoff, 1960), algoritma propagasi balik (Rumelhart dan McClelland, 1986).
2. *Unsupervised Learning* (proses belajar tak terawasi). Algoritma ini sama sekali tidak menggunakan data target (tanpa target). Kohonen (1984) mengembangkan pelatihan *unsupervised* (*unsupervised learning*). Pada algoritma belajar ini, tidak membutuhkan target untuk keluarannya.
3. *Reinforced Learning,* pada metode ini target tersedia, tetapi bukan sebagai penentu jawaban, hanya sebagai pengindikasi *output* yang dihasilkan benar atau salah. Metode tidak termasuk pembelajaran yang popular digunakan.

2.5 *Cluster Analisys*

Analisis *Cluster* adalah upaya menemukan sekelompok obyek yang mewakili suatu karakter yang sama atau hampir sama (*similar*) antar satu obyek dengan obyek lainnya pada suatu kelompok dan memiliki perbedaan (*not similar*) dengan obyek – obyek pada kelompok lainnya. Tentunya persamaan dan perbedaan tersebut diperoleh berdasar informasi yang diberikan oleh obyek – obyek tersebut beserta hubungan (*relationship*) antar obyek. Dalam berbagai

kesempatan, *clustering* juga sering disebut sebagai *Unsupervised Classification.* (Budhi, 2006)

2.6 *Min-max Normalization*

Metode normalisasi ini menghasilkan transformasi *linier* pada data asal. Bila *minA* dan *maxA* adalah nilai minimun dan maksimum dari sebuah atribut A, *Min-max Normalization* memetakan sebuah nilai v dari A menjadi v' dalam *range* nilai minimal dan maksimal yang baru, *new_minA* dan *new_maxA.* Rumus *Min-max Normalization* dapat dilihat pada persamaan 1. (Budhi, 2006)

$$v = \frac{v - minA}{maxA - minA} * (new\_maxA - new\_minA) + new\_minA \quad \ldots\ldots\ldots\ldots (2.1)$$

2.7 *Euclidean Distance*

Euclidean distance, yaitu mengukur jumlah kuadrat perbedaan nilai masing-masing variable.

$$d = \sqrt{\sum_{i}^{n} (Wi - Xi)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

Dimana:

d = jarak *Euclidian*

Wi= vektor bobot ke-i

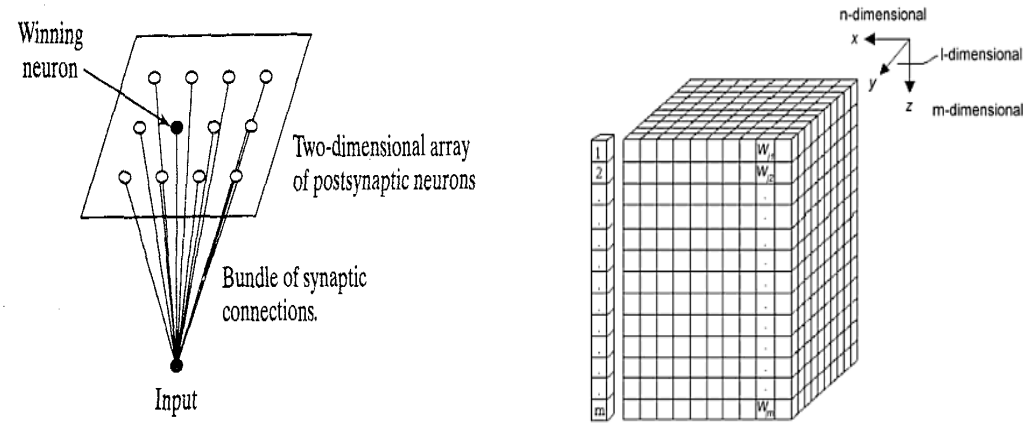
Xi= vektor *input* ke Xi

Semakin kecil nilai d, semakin besar kesamaan antara kedua obyek atau kasus tersebut, dan sebaliknya, semakin besar nilai d, semakin kecil kesamaan antara dua obyek.

2.8 *Kohonen*

Self-Organizing Maps (SOM), atau *Kohonen*, merupakan salah satu tipe dari *artifcial neural network* yang mana sistem pembelajarannya menggunakan *unsupervised learning* untuk menghasilkan dimensi rendah (pada umumnya dua dimensi). *Unsupervised learning* adalah *input* disini tidak terdefinisikan. model ini tidak membutuhkan *input*an yang selalu ada, sehingga tidak mempengaruhi *output*. Jadi jika *input* dari model ini hilang, tidak akan memepengaruhi variabel karena tidak memiliki target. Tipe ini berlawanan dengan *supervised learning* yang konsepnya adalah *input* diasumsikan diawal sedangkan *output* diakhir. Antara *input* dan *output* saling berhubungan sehingga kita pasti dapat mepredisikan *output* yang akan keluar (memiliki target). SOM ini ditemukan oleh Teuvo Kohonen (http://www.cis.hut._/harri/thesis/ valpola_thesis/node34.html).

Jaringan SOM (*Self-Organizing Map*) Kohonen merupakan salah satu model jaringan syaraf yang menggunakan metode pembelajaran *unsupervised*. Jaringan SOM Kohonen terdiri dari dua lapisan (*layer*), yaitu lapisan *input* dan lapisan *output*. Setiap neuron dalam lapisan *input* terhubung dengan setiap neuron pada lapisan *output*. Setiap neuron dalam lapisan *output* merepresentasikan kelas dari *input* yang diberikan (Madarum, 2003).



Gambar 2.4. Kohonen Model (Haykin, 2005)

Setiap neuron *output* mempunyai bobot untuk masing-masing neuron *input*. Proses pembelajaran dilakukan dengan melakukan penyesuaian terhadap setiap bobot pada neuron *output*. Setiap *input* yang diberikan dihitung jarak *Euclidian*-nya dengan setiap neuron *output*, kemudian cari neuron *output* yang mempunyai jarak minimum. Neuron yang mempunyai jarak yang paling kecil disebut neuron pemenang atau neuron yang paling sesuai dengan *input* yang diberikan. (Lihat rumus 2.2)

Setelah mendapatkan neuron pemenang maka update nilai bobot neuron pemenang dan tetangganya dengan perhitungan sebagai berikut:

$$W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)] \quad \ldots\ldots\ldots\ldots\ldots (2.3)$$

Dimana $0 < \alpha(t) < 1$

x = *input pixel*

w = bobot

t = waktu

i = *index node input*

j = *index node output*

α(t) adalah *learning rate* dan *h*(t) adalah pengaruh jarak neuron tetangga terhadap neuron pemenang pada pembelajaran.

$$h(t) = \exp\left(-\frac{||\boldsymbol{r}_b - \boldsymbol{r}_i||^2}{2\sigma^2(t)}\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.4)$$

Dimana : $\boldsymbol{r}_b$ dan $\boldsymbol{r}_i$ adalah posisi neuron pada SOM, σ adalah fungsi ketetanggaan.

Update tingkat pembelajaran dan radius ketetanggan menggunakan rumus sebagai berikut :

$$\alpha(t) = \alpha_0 \exp(-\frac{t}{\lambda}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.5)$$

$$\sigma(t) = \sigma_0 \exp(-\frac{t}{\lambda}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.6)$$

Dimana :

α_0 = tingkat pembelajaran pada t_0

σ_0 = radius ketetanggaan pada t_0

t = iterasi

λ = konstanta waktu

Secara garis besar algoritma SOM Kohonen sebagai berikut.

1. Inisialisasi, bobot (Wij) dengan nilai random, tingkat pembelajaran (*learning rate*) , dan fungsi tetangga.

2. Masukan *input* Xi
3. Hitung similaritasnya dengan menggunakan jarak *Euclidian*, dan pilih neuron pemenangnya.
4. *Update* bobot neuron pemenang dan tetangganya
5. *Update* tingkat pembelajaran dan kurangi fungsi tetangga.
6. Lakukan langkah 2 sampai 5 sampai nilai *epoch* tercapai.

2.9 Pemrograman Berorientasi Objek

Pemrograman Berorientasi Objek dimulai pertama kali dengan bahasa Simula yang dikembangkan di Scandinavia di pertengahan tahun 60-an. Simula utamanya digunakan untuk pemrograman simulasi, dimana adalah alamiah untuk memodelkan suatu entitas eksternal / diluar sistem perangkat lunak dan untuk memilih istilah-istilah untuk entitas-entitas tersebut dan tingkah lakunya. Simula memiliki sintak yang mirip dengan Pascal, tetapi programmer berfikir sedikit lebih berbeda ketika merancang suatu program yang akan dibuat dengan Simula. Sebuah ide dasar yang diperkenalkan dalam Simula adalah *inheritance* (pewarisan). Dalam Simula juga sudah dikenal objek (entitas) yang ada dalam sistem yang dimodelkan. Ada beberapa objek yang dikumpulkan kemudian disebut ”*Class*”, dan tugas utama dari seorang perancang program dengan Simula adalah menentukan *behaviour* dari *class* tersebut.

Setelah Simula, bahasa berikutnya yang diketahui mendukung untuk mengadopsi pemrograman berorientasi objek adalah *Smalltalk* yang

dikembangkan tahun 70-an di Xerox PARC. Simula dan *Smalltalk* dirancang secara elegan, dan menawarkan sejumlah konsep yang *powerful* yang memudahkan untuk dipelajari. (Somantri, 2004)

2.7.1. Abstraksi

Abstraksi adalah suatu cara melihat suatu objek dalam bentuk yang sederhana. Sebagai contoh jika kita melihat sepeda motor. Kita tidak perlu melihat susunan komponen mesin dan dukungan elektriknya yang cukup kompleks dan rumit, namun kita bisa melihat sepeda motor itu sebagai sebuah entitas / satuan tunggal (*single entity*) yang merupakan sebuah objek yang mempunyai sifat dan karakteristik tersendiri. Dengan pemikiran yang sederhana ini maka ketika kita mengendarai sepeda motor tersebut kita tidak perlu tahu betapa rumit komponen dan rangkaian yang menyusun sepeda motor. Karena untuk mengendarai sepeda motor yang perlu diketahui adalah bagaimana sepeda motor itu bisa dikendalikan.

Sehingga dengan konsep abstraksi ini kita bisa melihat suatu sistem yang kompleks yang terdiri dari subsistem-subsistem yang rumit dan banyak bisa dipandang menjadi sebuah paket sistem yang sederhana. (Somantri, 2004)

Pemahaman objek disekitar kita inilah yang akan mendasari pemahaman tentang pemrograman berorientasi objek. Yang paling penting adalah bagaimana mentransformasikan apa yang anda ketahui tentang suatu objek menjadi suatu program.

2.7.2. Elemen sebuah Objek: *state* dan *behaviour*

Setiap objek selalu memiliki *state* dan *behaviour* yang dapat mengubah *state* tersebut. Sebagai contoh manusia memiliki *state*: umur, tinggi, berat dan sebagainya. Demikian pula manusia memiliki *behaviour*: menua, meninggi, makan (menambah berat badan) dan sebagainya. *Behaviour* juga dapat tidak mengubah sama sekali *state* dari objek tersebut, baik secara langsung maupun tidak langsung. Secara sederhana *state* bisa dianggap sebagai suatu kata benda karena sifatnya yang pasif, benda tersebut tidak melakukan operasi tetapi justru menjadi target atau bahan operasi. Demikian pula *behaviour* bisa dianggap sebagai kata kerja, karena ia berfungsi untuk menujukkan operasi apa yang dilakukan. (Somantri, 2004)

2.7.3. Konsep Pemrograman Berorientasi Objek

Ciri khas pemrograman berorintasi objek adalah: Enkapsulasi, Pewarisan dan *Polymorphism*. (Somantri, 2004)

a. Enkapsulasi

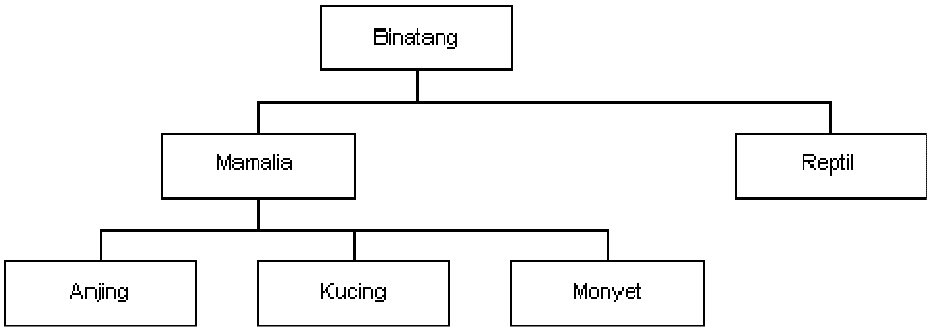
Enkapsulasi adalah suatu mekanisme untuk menyembunyikan atau memproteksi suatu proses dari kemungkinan interferensi atau penyalahgunaan dari luar sistem sekaligus menyederhanakan penggunaan system itu sendiri. Akses ke internal sistem diatur sedemikian rupa melalui seperangkat *interface*. Contoh kasus sepeda motor tadi, pada sistem pemindahan gigi transmisi, maka pengendara tidak perlu tahu detail dari bagaimana proses pemindahan gigi itu dilakukan oleh mesin, cukup tahu bagaimana menekan gigi transmisi itu. Pedal

gigi transmisi yang diinjak pengendara itu merupakan *interface* (antar muka) pengendara dengan sistem transmisi sepeda motor.

Dalam bahasa *Java* segala sesuatu adalah objek (*everything is object*). Setiap baris program yang ditulis *programmer* adalah merupakan bagian dari sebuah objek. *Programmer* juga dapat membangun sebuah objek yang disusun oleh objek-objek kecil, dimana masing-masing objek yang menyusunnya memiliki fungsi sendiri-sendiri.

b. Pewarisan (*Inheritance*)

Sebagai manusia kita sebenarnya terbiasa untuk melihat objek yang berada disekitar kita tersusun secara hierarki berdasarkan *class*-nya masing-masing. Dari sini kemudian timbul suatu konsep tentang pewarisan yang merupakan suatu proses dimana suatu *class* diturunkan dari *class* lainnya sehingga ia mendapatkkan ciri atau sifat dari *class* tersebut. Perhatikan contoh hirarki berikut ini:



Dari hirarki diatas dapat dilihat bahwa, semakin kebawah, *class* akan semakin bersifat spesifik. *Class* mamalia memiliki seluruh sifat yang dimiliki oleh binatang, demikian halnya juga Anjing, kucing dan Monyet memiliki seluruh sifat yang diturunkan dari *class* mamalia. Dengan konsep ini, karakteristik yang

dimiliki oleh *class* binatang cukup didefinisikan didefinisikan dalam *class* binatang saja. *Class* mamalia tidak perlu mendefinisikan ulang apa yang telah dimiliki oleh *class* binatang, karena sebagai *class* turunannya, ia akan mendapatkan karakteristik dari *class* binatang secara otomatis. Demikian juga dengan *class* anjing, kucing dan monyet, hanya perlu mendefinisikan karakteristik yang spesifik dimiliki oleh *class*-nya masing-masing. Dengan memanfaatkan konsep pewarisan ini dalam pemrograman, maka hanya perlu mendefinisikan karakteristik yang lebih umum akan didapatkan dari *class* darimana ia diturunkan.

c. Polymorphism

Polymorphism berasal dari bahasa Yunani yang berarti banyak bentuk. Dalam PBO, konsep ini memungkinkan digunakannya suatu *interface* yang sama untuk memerintah objek agar melakukan aksi atau tindakan yang mungkin secara prinsip sama namun secara proses berbeda. Dalam konsep yang lebih umum sering kali *Polymorphism* disebut dalam istilah satu *interface* banyak aksi. Contoh yang konkrit dalam dunia nyata yaitu mobil. Mobil yang ada dipasaran terdiri atas berbagai tipe dan berbagai merk, namun semuanya memiliki *interface* kemudi yang sama, seperti: stir, tongkat transmisi, pedal gas dan rem. Jika seseorang dapat mengemudikan satu jenis mobil saja dari satu merk tertentu, maka orang itu akan dapat mengemudikan hamper semua jenis mobil yang ada, karena semua mobil tersebut menggunakan *interface* yang sama. Harus diperhatikan disini bahwa *interface* yang sama tidak berarti cara kerjanya juga sama. Misalnya pedal

gas, jika ditekan maka kecepatan mobil akan meningkat, tapi bagaimana proses peningkatan kecepatan ini dapat berbeda-beda untuk setiap jenis mobil.

2.10 *Unified Modeling Language*

Unified Modeling Language (UML) merupakan standar dalam menentukan visualisasi, konstruksi, dan mendokumentasikan *artifacts* dari sistem *software,* untuk memodelkan bisnis, dan sistem non *software* lainnya. *Artifacts* adalah sepotong informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa *software*, *artifacts* dapat berupa model, deskripsi, atau *software* (Sutopo, 2002).

UML merupakan perkembangan dari metode-metode perancangan yang sebelumnya seperti:

1. Metode *Booch*, dikembangkan oleh Grady Booch (1994). Karakteristik: *infamous clouds hard to draw* (Sutopo, 2002).
2. *Object Modelling Techinque* (OMT), dikembangkan oleh James Rumbaugh(1991). Karakteristik: sangat mirip dengan UML (Sutopo, 2002).
3. *Object Oriented System Engineering (OOSE)/Objectory Important Contribution Uses Case*, dikembangkan oleh Ivar Jacobson (1992).

2.8.1. Tujuan UML

Tujuan utama UML diantaranya adalah untuk:

a. Memberikan model yang siap pakai, bahasa pemodelan *visual* yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum
b. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.

2.8.2. Pembagian *Views*

UML terdiri dari beberapa diagram, diantaranya yaitu: (Nugroho, 2004):

1. *Use Case Views*

Model ini berfungsi untuk menggambarkan sistem *design outside user* (pemakai luar) yang disebut *actor*. Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai dengan yang diinginkan *external actors*. *Actor* yang berinteraksi dengan sistem dapat berupa *user* atau sistem lainnya. *View* ini digambarkan dalam *use case diagrams* dan kadang-kadang dengan *activity diagrams*. *View* ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

Diagram *use case* digunakan untuk menggambarkan hubungan transaksi antara sistem dan *end user*, selain itu diagram *use case* dapat diartikan sebagai gambaran *actor* dengan kumpulan *use case* yang menyertakan batasan sistem, kumpulan komunikasi antara *actor* dan *use case* dan generalisasi diagram *use case* (Suhendar, 2002).

Simbol-simbol yang terdapat di dalam diagram *use case* yaitu:

a. *Use case* adalah spesifikasi rangkaian dari tindakan baik termasuk rangkaian berbeda ataupun rangkaian salah yang sistem, subsistem atau *class* dapat dilakukan dari hubungan dengan pihak luar. *Use case* adalah gambaran fungsionalitas dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

Cara menentukan *Use case* dalam suatu sistem:

1. Pola perilaku perangkat lunak aplikasi.
2. Gambaran tugas dari sebuah *actor*.
3. Sistem atau "benda" yang memberikan sesuatu yang bernilai kepada *actor*.
4. Apa yang dikerjakan oleh suatu perangkat lunak (bukan bagaimana cara mengerjakannya).

Gambar 2.6. Simbol *Use Case*

b. *Actor*

Actor adalah abstraksi untuk *entity* luar dari sistem, subsistem atau *class* yang berhubungan secara langsung dengan sistem. Pada *use diagram* diperlukan beberapa *actor* dimana *actor* tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem.

Sebuah *actor* mungkin hanya memberikan informasi *input*an pada sistem, hanya menerima informasi dari sistem atau keduanya menerima dan memberi informasi pada sistem, *actor* hanya berinteraksi dengan *use case* tetapi tidak memiliki control atas *use case*. *Actor* digambarkan dengan *stick man*. *Actor* dapat digambarkan secara umum atau spesifik, dimana untuk membedakannya dapat menggunakan *relationship*.

Ada beberapa kemungkinan yang menyebabkan *actor* tersebut terkait dengan sistem antara lain:

1. Yang berkepentingan terhadap sistem dimana adanya arus informasi baik yang diterimanya maupun yang di*input*kan ke sistem.
2. Orang atau pihak yang akan mengelola sistem tersebut.
3. *External resource* yang digunakan oleh sistem.
4. Sistem lain yang berinteraksi dengan sistem yang akan dibuat.

Gambar 2.7. Simbol *Actor*

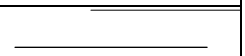
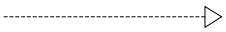
2. *Static Views*

Static Views adalah gambaran tentang keseluruhan model yang digolongkan berdasarkan sistem dan hubungan statis. Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class*, *object*, dan *relationship*) dan

kolaborasi dinamis yang terjadi ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu. *View* ini digambarkan dalam *class diagrams* untuk struktur statis dan dalam *state*, *sequence*, *collaboration*, dan *activity diagram* untuk model dinamisnya. *View* ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

Relationship adalah hubungan semantik antara model elemen. *Relationship* didalam UML bermacam-macam, diantaranya yaitu:

Tabel 2.1. Macam-macam *Relationship* (Relasi)

Relationship	***Function***	***Notation***
Association	Menghubungkan *link* antar elemen	
Generalization	Sebuah elemen dapat merupakan spesialisasi dari elemen lainnya. Digunakan untuk pewarisan (*inheritance*)	
Depedency	Hubungan diantara dua model elemen	
Realization	Hubungan diantara spesifikasi dan implementasinya.	
Aggregation	Bentuk *association* dimana sebuah elemen berisi elemen lainnya.	

3. *Interaction Views*

Interaction Views mendeskripsikan tentang rangkaian dari pertukaran perintah berdasarkan peraturan yang merupakan jalan dari pelaksanaan sistem.

Interaction Views ini digambarkan dengan menggunakan *sequence diagram.* (Bahrami, 1999).

4. *State Machine Views*

State machine views adalah model tentang sejarah dari objek dan kelas. *State machine* mengandung *state* yang dihubungkan dengan transisi. *State machine views* digambarkan dengan *state chart diagram.*

5. *Activity Views*

Activity graph berbeda dengan *state machine* yang menggambarkan perhitungan aktivitas termasuk didalam pelaksanaan perhitungan. *Activity state* mewakili aktivitas langkah aliran kerja atau eksekusi dari sebuah operasi.

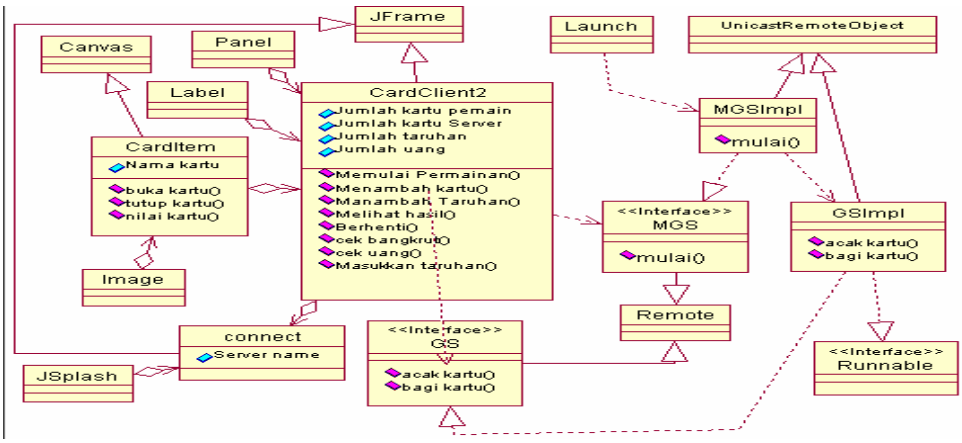
6. *Deployment View*

Deployment View memperlihatkan pemetaan setiap proses ke dalam *hardware, view* ini paling bermanfaat ketika membuat model suatu sistem yang diterapkan dalam lingkungan arsitektural yang terdistribusi yakni ketika menerapkan aplikasi dan *server* pada lokasi yang berbeda. *View* ini hanya memiliki satu diagram, yaitu *deployment diagram.*

Ada delapan diagram yang digunakan untuk pemodelan objek dan kelas dengan menggunakan bahasa UML: (Nugroho, 2004)

A. *Class Diagram*

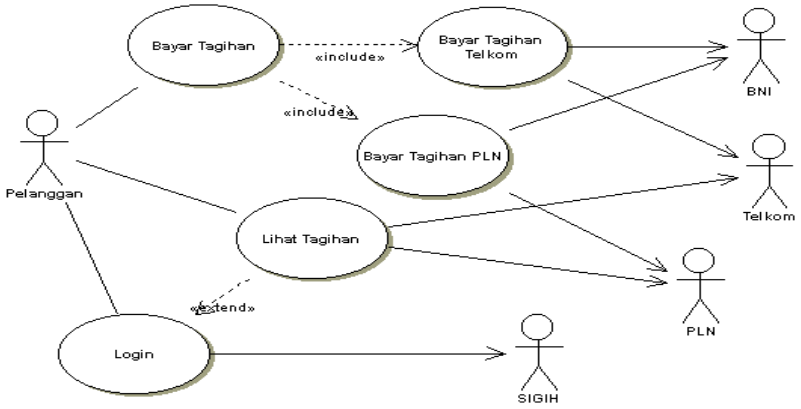
Digunakan untuk menampilkan hubungan antar kelas-kelas dan informasi mengenai suatu kelas. *Class diagram* adalah dasar dari *component diagram* dan *deployment diagram*.



Gambar 2.8. Contoh *Class Diagram*

B. *Uses-case*

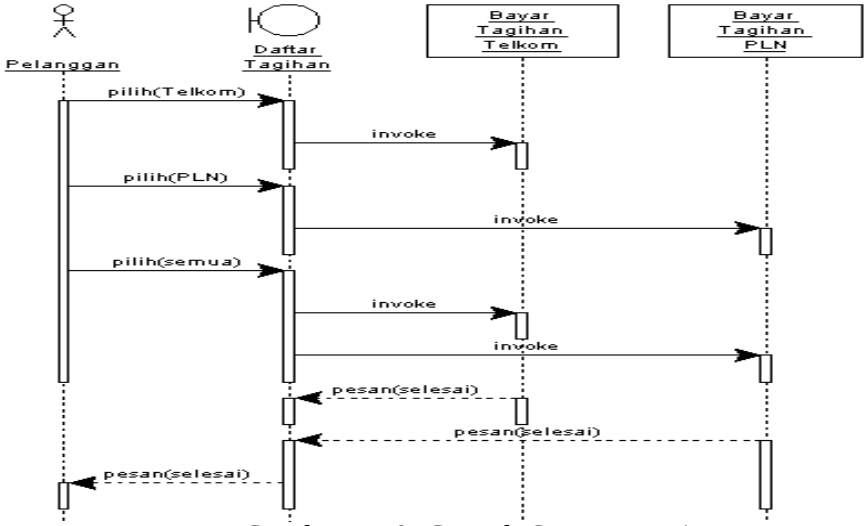
Digunakan untuk memperlihatkan sistem secara garis besar *high level view*, terutama dilihat dari perspektif pengguna.



Gambar 2.9. Contoh *Use Case Diagram*

C. *Sequence Diagram*

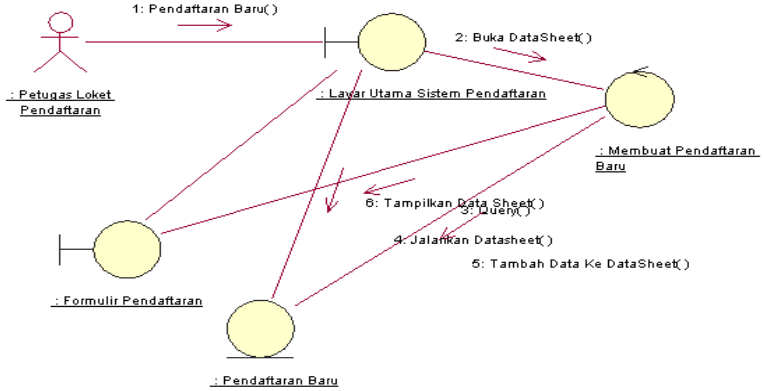
Digunakan untuk menggambarkan interaksi objek-objek berdasarkan urut-urutan waktu (*time sequence*). Berikut adalah contoh penggunaan dari *sequence diagram:*



Gambar 2.10. Contoh *Sequence Diagram*

D. *Collaboration Diagram*

Digunakan untuk menggambarkan interaksi atau hubungan struktural antar objek-objek dalam model. Berikut adalah contoh penggunaan dari *collaboration diagram*:

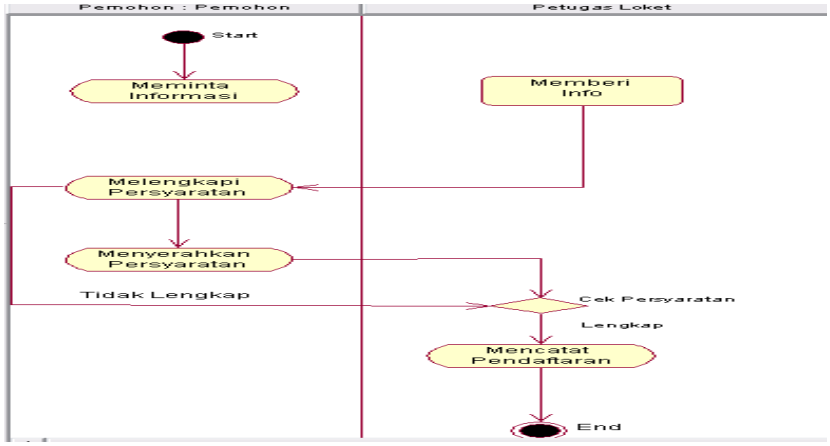


Gambar 2.11. Contoh *Collaboration Diagram*

E. *Activity Diagram*

Digunakan untuk memodelkan alur kerja (*workflow*) sebuah proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini

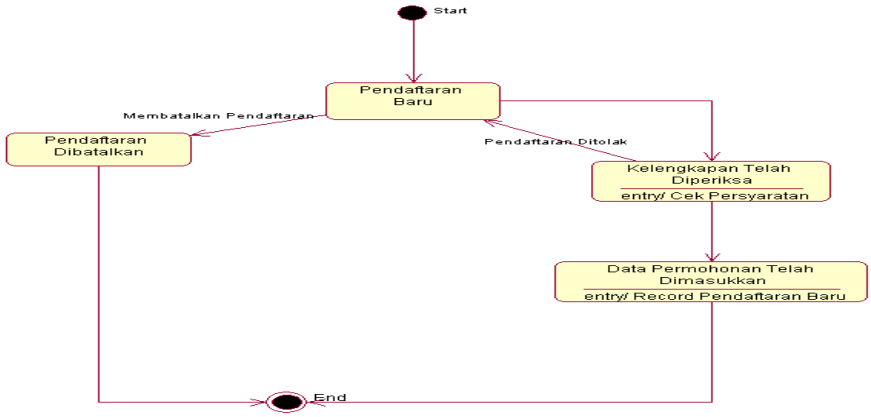
sangat mirip dengan *flowchart.* Berikut adalah contoh penggunaan dari *activity diagram*:



Gambar 2.12**.** Contoh *Activity Diagram*

F. *Statechart Diagram*

Diagram ini menunjukkan urut-urutan *state* yang dilalui sebuah objek, *event* yang menyebabkan perpindahan dari suatu *state* ke *state* yang lain dan aksi-aksi yang dilakukan pada sebuah *state*. Diagram ini biasanya digunakan untuk memodelkan tahap-tahap dari kehidupan sebuah objek. Berikut adalah penggunaan contoh *statechart diagram*:



Gambar 2.13**.** Contoh *Statechart Diagram*

2.7 *Java*

Java merupakan bahasa pemrograman berorientasi objek dan bebas *platform*, yang dikembangkan oleh seorang insinyur *SUN Micro system* bernama James Gosling yang pada awalnya diberi nama Oak dan diubah menjadi *Java* dengan sejumlah keunggulan yang memungkinkan *Java* dijadikan sebagai bahasa pengembangan *enterprise*.

2.7.1 Fitur – Fitur *Java* yang Menarik

Beberapa fitur yang ditawarkan *Java* API antara lain sebagai berikut :

a. *Applet*

Program *Java* yang dapat berjalan di atas *browser*, yang dapat membuat halaman HTML lebih dinamis dan menarik.

b. *Java Networking*

Sekumpulan API (*Application Programming Interface*) yang menyediakan fungsi – fungsi untuk aplikasi – aplikasi jaringan, seperti penyediaan akses untuk TCP, UDP, IP Adrress dan URL. Tetapi *Java Networking* tidak menyediakan akses untuk ICMP dikarenakan alasan sekuriti dan pada kondidi umum hanya *administrator* (*root*) yang bisa memanfaatkan protokol ICMP.

c. *Java Database Connectivity* (JDBC)

JDBC menyediakan sekumpulan API yang dapat digunakan untuk mengakses database seperti *Oracle, MySQL, PostgreSQL, Microsoft SQL Server*.

d. *Java Security*

Java Security menyediakan sekumpulan API untuk mengatur *security* dari aplikasi *Java* baik secara *high level* atau *low level*, seperti *public/private key management* dan *certificates*.

e. *Java Swing*

Java Swing menyediakan sekumpulan API untuk membangun aplikasi – aplikasi GUI (*Graphical User Interface*) dan model GUI yang diinginkan bisa bermacam – macam, bisa model *Java*, model Motif/CDE atau model yang dependent terhadap *platform* yang digunakan.

f. *Java* RMI

Java RMI menyediakan sekumpulan API untuk membangun aplikasi – aplikasi *Java* yang mirip dengan model RPC (*Remote* 4 *Procedure Call*) jadi object - object *Java* bisa di *call* secara *remote* pada jaringan komputer.

g. *Java* 2D/3D

Java 2D/3D menyediakan sekumpulan API untuk membangun grafik – grafik 2D/3D yang menarik dan juga akses ke *printer*.

h. *Java Server Pages*

Berkembang dari *Java Servlet* yang digunakan untuk menggantikan aplikasi – aplikasi CGI, JSP (*Java Server Pages*) yang mirip ASP dan PHP merupakan alternatif terbaik untuk solusi aplikasi *Internet*.

i. JNI (*Java Native Interface)*

JNI menyediakan sekumpulan API yang digunakan untuk mengakses fungsi – fungsi pada *library* (*.dll atau *.so) yang dibuat dengan bahasa pemrograman yang lain seperti C,C++, dan *Basic*.

j. *Java Sound*

Java Sound menyediakan sekumpulan API untuk manipulasi *sound*.

k. *Java* IDL + CORBA

Java IDL (*Interface Definition Language*) menyediakan dukungan *Java* untuk implementasi CORBA (*Common Object Request Broker*) yang merupakan model *distributed-Object* untuk solusi aplikasi besar di dunia *networking*.

l. *Java Card*

Java Card utamanya digunakan untuk aplikasi – aplikasi pada *smart card*, yang sederhana wujudnya seperti SIM *Card* pada *handphone*.

m. JTAPI (*Java Telephony* API)

Java Telepony API menyediakan sekumpulan API untuk memanfaatkan *devices – devices telepony*, sehingga akan cocok untuk aplikasi – aplikasi CTI (*Computer Telephony Integration*) yang dibutuhkan seperti ACD (*Automatic Call Distribution*), PCPBX dan lainnya.

(sumber: www.asephs.web.ugm.ac.id/.../MODUL%20PEMROGRAMAN%20JAVA/.../*BAB%20I%20PENDAHULUAN.pdf*)

2.8 *Matlab*

MATLAB (*Matrix Laboratory*) adalah sebuah program untuk analisis dan komputasi numerik dan merupakan suatu bahasa pemrograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunkan sifat dan bentuk matriks. Pada awalnya, program ini merupakan *interface* untuk koleksi rutin-rutin *numeric* dari proyek LINPACK dan EISPACK, dan dikembangkan menggunkan bahasa FORTRAN namun sekarang merupakan produk komersial dari perusahaan *Mathworks, Inc*. yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan *assembler* (utamanya untuk fungsi-fungsi dasar MATLAB).

MATLAB telah berkembang menjadi sebuah *environment* pemrograman yang canggih yang berisi fungsi-fungsi *built-in* untuk melakukan tugas pengolahan sinyal, aljabar *linier*, dan kalkulasi matematis lainnya. MATLAB juga berisi *toolbox* yang berisi fungsifungsi tambahan untuk aplikasi khusus . MATLAB bersifat *extensible*, dalam arti bahwa seorang pengguna dapat menulis fungsi baru untuk ditambahkan pada *library* ketika fungsi-fungsi *built-in* yang tersedia tidak dapat melakukan tugas tertentu. Kemampuan pemrograman yang dibutuhkan tidak terlalu sulit bila Anda telah memiliki pengalaman dalam pemrograman bahasa lain seperti C, PASCAL, atau FORTRAN.

MATLAB merupakan merk software yang dikembangkan oleh *Mathworks.Inc.* (lihat http://www.mathworks.com) merupakan *software* yang paling efisien untuk perhitungan *numeric* berbasis matriks. Dengan demikian jika

di dalam perhitungan kita dapat menformulasikan masalah ke dalam format matriks maka MATLAB merupakan *software* terbaik untuk penyelesaian numericnya.

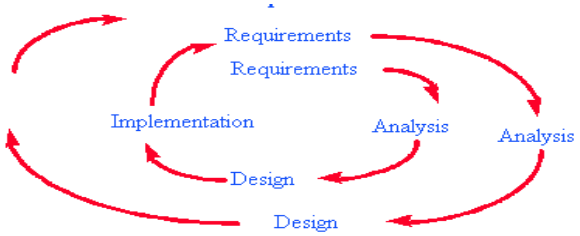
MATLAB (*Matrix Laboratory*) yang merupakan bahasa pemrograman tingkat tinggi berbasis pada matriks sering digunakan untuk teknik komputasi numerik, yang digunakan untuk menyelesaikan masalah-masalah yang melibatkan operasi matematika elemen, matrik, optimasi, *aproksimasi* dll. Sehingga Matlab banyak digunakan pada :

1. Matematika dan Komputansi
2. Pengembangan dan Algoritma
3. Pemrograman modeling, simulasi, dan pembuatan prototype
4. Analisa Data , eksplorasi dan visualisasi
5. Analisis numerik dan statistic
6. Pengembangan aplikasi teknik

(sumber : *elista.akprind.ac.id/upload/files/4544_Modul2.pdf*)

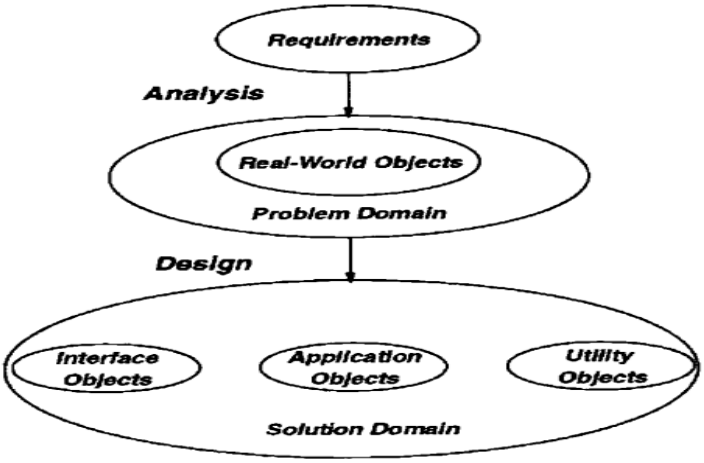
2.9 Metodologi Pengembangan Sistem (*Software Development Life Cycle)*

Didalam *Software Development Life Cycle* (SDLC) ditentukan fase-fase pembangunan suatu *software,* mulai dari Initiation, *Software Concept, Planning, Requirements Analysis, Design, Development (Coding), Testing, Implementation*, dan lain-lain. *Object Oriented* dapat diadaptasikan pada setiap model pengembangan sistem, tetapi pilihan terbaik adalah model proses evolusioner (Pressman, 1997).



Gambar 2.14. Model proses OO (Pressman, 1997)

Pada penelitian ini penegembangan sistem menggunakan OOAD. OOAD adalah *Object Oriented Analysis and Design*. OOAD juga sebuah metodologi dan merupakan subset dari SDLC. Di dalam OOAD ditentukan metode-metode yang harus dilakukan pada tahap *Software Concept, Requirements Analysis*, dan *Design*. OOAD memahami permasalahan dan solusi *logic* dari sudut pandang *object* (benda, konsep, entitas). Pemodelan dilaksanakan dengan penggunaan *Unified Modelling Language (UML).*



Gambar 2.15**.** *Object Oriented Analysis and Design* (OOAD)

BAB III

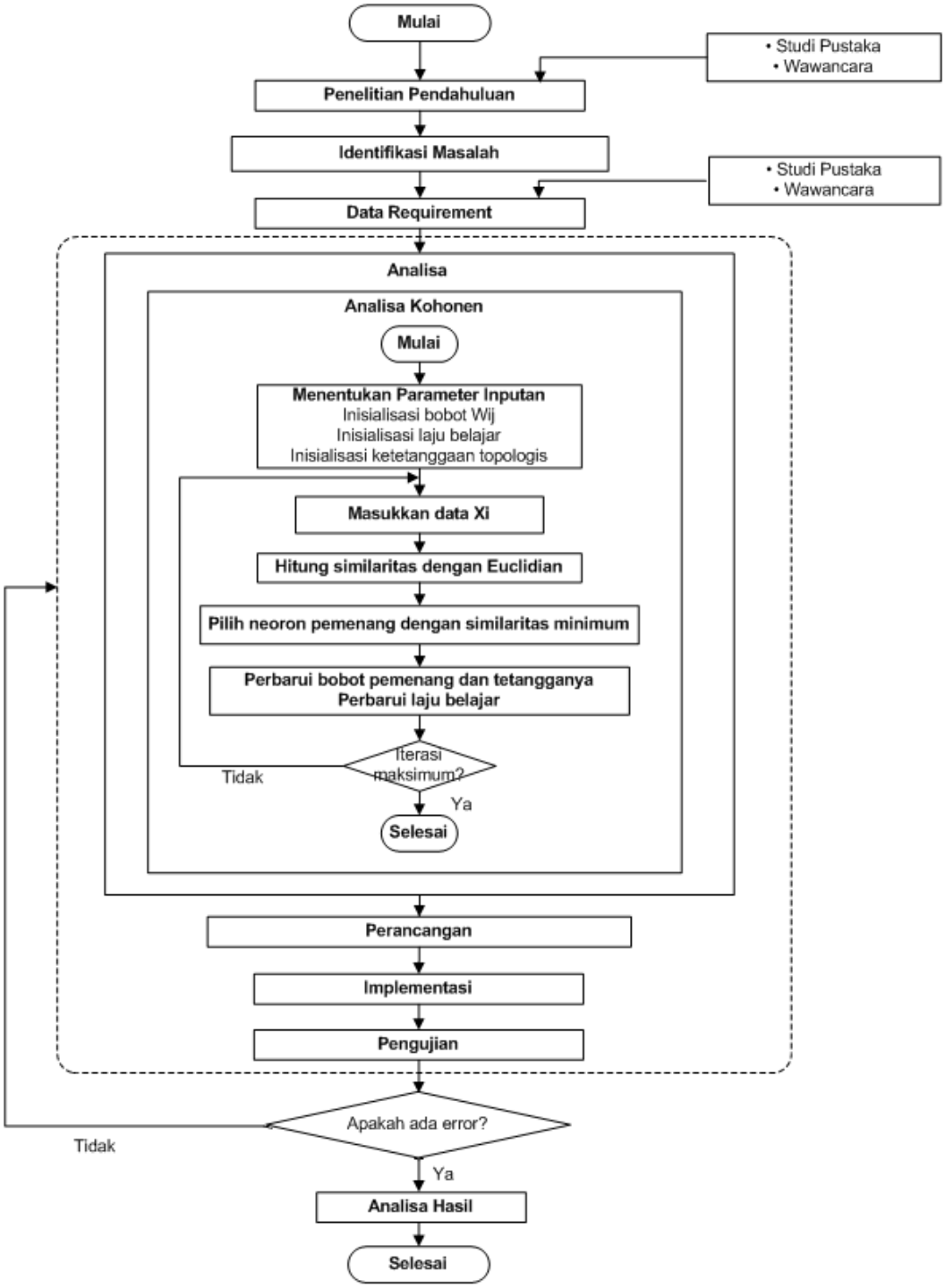
METODOLOGI PENELITIAN

3.1 Data Penelitian

Data yang digunakan merupakan data sekunder yang diperoleh dari Tugas Akhir Sumini Iriani (2007). Dengan jumlah data sebanyak 50 buah data dan memiliki *field* WellID, Minyak, Air, Gas dan Tanggal. WellID merupakan sumur minyak yang memiliki satuan *barrel.* Pada data gas semuanya berjumlah 0, sehingga hanya data minyak dan air yang dijadikan masukkan untuk melakukan pengelompokkan daerah produksi minyak. Data selengkapnya terdapat pada lampiran A.

3.2 Tahapan Penelitian

Metodologi penelitian digunakan sebagai pedoman dalam pelaksanaan penelitian agar hasil yang dicapai tidak menyimpang dari tujuan yang telah dilakukan sebelumnya. Metodologi penelitian yang digunakan dalam penyusunan Tugas Akhir ini akan melalui beberapa tahapan yang membentuk sebuah alur yang sistematis Tahap-tahap yang akan dilalui digambarkan dengan *flowchart* berikut ini:



Gambar 3.1. *Flowchart* Tahapan Penelitian

3.2.1 Penelitian Pendahuluan

Pada tahapan ini, peneliti mengidentifikasi penelitian yang akan dilakukan. Berikut merupakan aktifitas yang dilaksanakan dalam penelitian pendahuluan :

a. Studi Pustaka

   Studi pustaka dilakukan dengan tujuan untuk mengetahui metode apa yang akan digunakan untuk menyelesaikan permasalahan yang akan diteliti, serta mendapatkan dasar-dasar referensi yang kuat dalam menerapkan suatu metode yang akan digunakan dalam Tugas Akhir ini Studi kepustakaan juga melihat serta membandingkan penelitian-penelitian yang sudah ada, sehingga peneliti mendapatkan tema penelitian mengenai pengelompokkan daerah produksi minyak bumi menggunakan metode *Kohonen*.

b. Wawancara

   Wawancara berfungsi untuk mengumpulkan informasi yang akan berguna dalam penelitian pengelompokkan daerah produksi minyak bumi menggunakan metode *Kohonen*. Wawancara dilakukan terhadap orang yang berkecimpung dalam bidang perminyakan.

3.2.2 Identifikasi Masalah

Setelah dilakukan penelitian pendahuluan diketahui bahwa perusahaan minyak memerlukan adanya pengelompokkan sumur minyak untuk mengetahui tingkat produksi suatu sumur minyak.

3.2.3 Data Requirements

Untuk mendapatkan data-data yang dibutuhkan, maka dilakukan beberapa metode pengumpulan data, yaitu:

a. Studi Pustaka

   Studi pustaka untuk mendapatkan dasar-dasar referensi yang kuat dalam menerapkan suatu metode yang akan digunakan dalam Tugas Akhir ini, yaitu dengan mempelajari Tugas Akhir Sumini Iriani (2007), buku-buku dan artikel-artikel yang berhubungan dengan permasalahan yang akan dibahas. Referensi yang digunakan penulis dalam menyelesaikan tugas akhir ini adalah Tugas Akhir Sumini Iriani (2007) dan yang berhubungan dengan jaringan saraf tiruan dan Kohonen (SOM). Penjelasan tentang referensi yang digunakan lebih lengkapnya dapat dilihat pada daftar pustaka.

b. Wawancara

   Wawancara (*interview*) adalah pengumpulan data dengan mengajukan pertanyaan-pertanyaan secara langsung kepada responden. Wawancara dalam hal ini menghasilkan data sekunder atau sebagai data pendukung data primer dalam melakukan pengolahan data. Wawancara yang akan dilakukan yakni wawancara terhadap orang yang berkecimpung dibidang perminyakan.

3.2.4 Analisa

Setelah melakukan penelitian pendahuluan, identifikasi masalah dan data *requirement*, kemudian langkah berikutnya adalah analisa kebutuhan perangkat lunak

Proses analisa perangkat lunak dilakukan untuk pemeriksaan masalah dan penyusunan alternatif pemecahan masalah yang timbul serta membuat spesifikasi sistem yang baru atau sistem yang akan diusulkan dan dimodifikasi. Analisa kebutuhan perangkat lunak terdiri dari:

1. Analisa Model Permasalahan

   Permasalahan yang akan diselesaikan adalah bagaimana merancang dan membangun suatu aplikasi jaringan syaraf tiruan untuk melakukan pengelompokkan daerah produksi minyak bumi menggunakan metode *Kohonen*.

2. Kebutuhan Data

   Aplikasi ini membutuhkan beberapa data masukan yaitu data minyak dan data air. Data tersebut didasarkan pada data tugas akhir Sumini Iriani (2007).

3. Analisa Penyelesaian Masalah

   Menganalisa data masukan, proses dan data keluaran dari aplikasi.

4. Analisa *Kohonen*

   Menganalisa metode *Kohonen* untuk permasalahan pengelompokkan daerah produksi minyak bumi.

3.2.5 Perancangan

Tahap perancangan sistem merupakan tahapan dalam membuat rincian sistem hasil dari analisis menjadi suatu bentuk perancangan agar dimengerti oleh pengguna (*user*).

a. Perancangan sistem seperti perancangan fungsi-fungsi yang akan digunakan dalam program aplikasi dari metode *Kohonen* dalam bentuk algoritma.
b. Perancangan UML (*unified modeling language*) yang meliputi *use case diagram, class diagram, sequence diagram, collaboration diagram, activity diagram, statechart diagram* dan *deployment diagram*.
c. Perancangan *interface* yang akan digunakan pada program aplikasi.

3.2.6 Implementasi

Pada tahap ini akan dikembangkan suatu sistem pengelompokkan daerah produksi minyak bumi dengan menggunakan bahasa pemrograman *Java*.

Mekanisme pembelajaran yang digunakan dalam penelitian ini adalah jaringan syaraf tiruan *Kohonen*. Kemudian akan dilakukan pengujian terhadap implementasi tersebut dan peninjauan kembali hasil dari kinerja sistem yang telah dikembangkan.

3.2.7 Pengujian

Tahap pengujian dilakukan dengan tujuan untuk menjamin sistem yang dibuat sesuai dengan hasil analisis dan perancangan serta menghasilkan satu kesimpulan apakah sistem tersebut sesuai dengan yang diharapkan.

Lingkungan pengujian :

a. Perangkat lunak dan sistem operasi yang digunakan dalam pengujian aplikasi menggunakan *Matlab* dan *Window*s XP.
b. Perangkat keras yang digunakan dalam pengujian aplikasi ini adalah komputer dengan spesifikasi:
    1. Prosesor Intel Core 2 Duo 1.66 GHz
    2. Memori 1 GB

Data pengujian :

a. Pengujian dilakukan terhadap data yang telah dilatihkan sebelumnya
b. Pengujian dilakukan juga pada data yang belum pernah dilatihkan sebelumnya.

Tujuan dilakukan pengujian terhadap aplikasi yang telah dibuat untuk membuktikan bagaimana akurasi aplikasi yang dirancang dengan metode jaringan syaraf tiruan *Kohonen.* Akurasi dihitung dengan pengujian *white box*, aplikasi dan *Matlab.*

3.2.8 Analisa Hasil

Menganalisa hasil pengujian yang telah dilakukan untuk mengetahui apakah sistem yang dibuat dapat mengelompokkan daerah produksi minyak bumi dengan baik atau tidak, dan bagaimana akurasi sistem dapat melakukan pengelompokkan.

BAB IV

ANALISA DAN PERANCANGAN

Analisa sistem bisa didefinisikan sebagai “Penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya “ [Jogiyanto, 1999]

Perancangan sistem adalah membuat rincian sistem hasil dari analisa menjadi bentuk perancangan agar dimengerti oleh pengguna.

4.1. Analisa Model Permasalahan

Permasalahan yang akan diselesaikan adalah bagaimana merancang dan membangun suatu aplikasi jaringan syaraf tiruan untuk melakukan pengelompokkan daerah produksi minyak bumi menggunakan metode *Kohonen*. Dalam penelitian ini pengelompokkan yang dilakukan membagi daerah produksi menjadi 3 *cluster*, ditampilkan dalam bentuk visualisasi tiga dimensi yang dibuat dalam bentuk *bubblechart*. Hasil *cluster* kemudian di analisa oleh *user* untuk menentukan kelompok dari *cluster* tersebut yaitu penuh, kering atau separuh penuh.

4.2. Kebutuhan Data

Data yang digunakan dalam pengamatan adalah data produksi sumur minyak bumi. Data yang digunakan merupakan data yang diambil dari Tugas Akhir Sumini Iriani (2007) . Data pengamatan selengkapnya terdapat pada lampiran A.

4.3. Analisa Kebutuhan Perangkat Lunak

Proses analisa perangkat lunak dilakukan untuk pemeriksaan masalah dan penyusunan alternatif pemecahan masalah yang timbul serta membuat spesifikasi sistem yang baru atau sistem yang akan diusulkan dan dimodifikasi

4.3.1. Analisa Data Masukan (*Input*)

Data yang akan dianalisa dalam aplikasi pengelompokkan daerah produksi minyak bumi adalah data masukan dan data keluaran. Adapun data masukan yang dibutuhkan untuk memulai pembuatan aplikasi ini adalah sebagai berikut:

1. Data produksi sumur minyak

   Data yang digunakan merupakan data sekunder yang diperoleh dari Tugas Akhir Sumini Iriani (2007). Dengan jumlah data sebanyak 50 buah data dan memiliki *field* WellID, Minyak, Air, Gas dan Tanggal. WellID merupakan sumur minyak yang memiliki satuan *barrel*. Pada data gas semuanya berjumlah 0, sehingga hanya data minyak dan air yang dijadikan masukkan untuk melakukan pengelompokkan daerah produksi minyak.

2. Jumlah iterasi

   Jumlah iterasi merupakan banyaknya *loop program* yang dilakukan pada algoritma *kohonen*. Pada kasus ini jumlah iterasi diinputkan oleh user.

3. Jumlah *cluster*

   Jumlah *cluster* merupakan inputan yang menentukan banyaknya jumlah kelompok yang akan dihasilkan. Pada kasus ini *cluster* berjumlah 3 yang mewakili dari sumur minyak yang penuh, kering dan setengah penuh. Pemilihan 3 *cluster* berdasarkan masukkan dari salah seorang pegawai yang bekerja di perusahaan minyak sekaligus dosen luar biasa, Bapak Lilik Trihardianto, S.Si. Hal ini dikarenakan pada perusahaan minyak belum ada aturan baku yang menetapkan pembagian kelompok daerah produksi minyak bumi.

4. Laju pembelajaran (*Learning rate*)

   Digunakan untuk menunjukkan bagaimana adaptasi pembelajaran terhadap data. Pada aplikasi ini nilai laju pembelajaran diinputkan oleh *user*. Nilai yang dipilih adalah $0 \leq \alpha(t) \leq 1$, karena semakin dekat learning rate $\alpha(t)$ mendekati 0, vektor-vektor input dapat dipetakan dengan baik. (Budhi, 2006)

5. Radius ketetanggaan (*Neighborhood Function*)

   Pada aplikasi ini radius ketetanggaan yang digunakan adalah *Cubic Topology*, karena menggunakan visualisasi 3D (3 Dimensi). Radius Ketetanggaan berfungsi memberi pengaruh perubahan bobot secara proporsional dari neuron best matching ke *neuron - neuron* tetangganya.

4.3.2. Analisa Proses

Dari data-data masukan yang diperoleh sebelumnya, proses pengelompokkan bekerja setelah aplikasi memberikan data yang telah diinputkan oleh *user.*

Langkah-langkah yang terjadi dalam proses pengelompokkan daerah produksi minyak bumi secara garis besarnya adalah sebagai berikut:

Langkah I *User* menginputkan data masukkan ke dalam aplikasi

Langkah II Aplikasi melakukan pengelompokkan dari data masukkan yang telah diinputkan oleh *user.* Pengelompokkan data dilakukan dengan menggunakan metode *Kohonen.*

Langkah III Hasil dari proses pengelompokkan kemudian ditampilkan dalam bentuk visualisasi *bubblechart*.

4.3.3. Analisa Keluaran (*Output*)

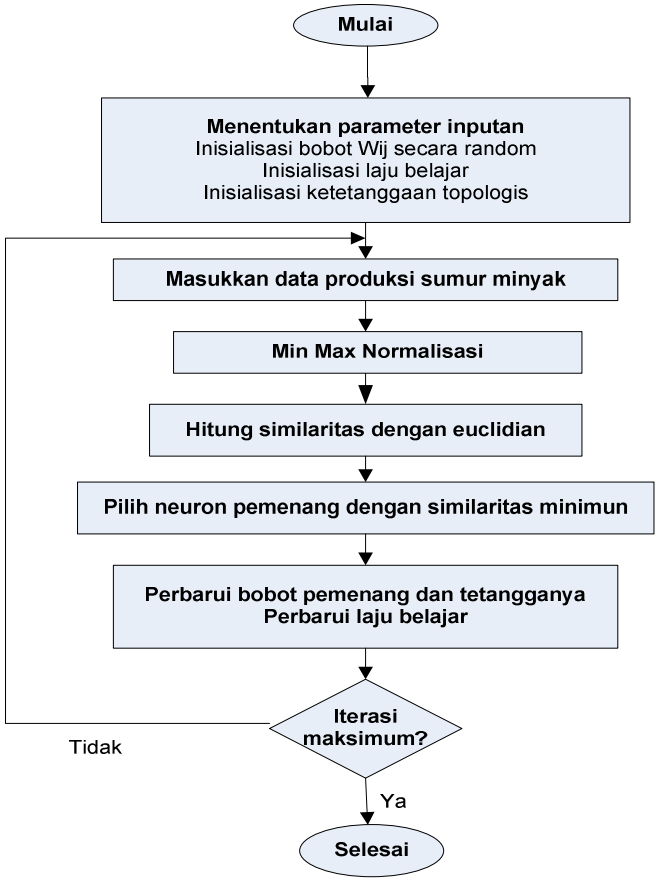
Tujuan akhir dari pembuatan aplikasi ini adalah pengelompokkan daerah produksi minyak bumi berdasarkan tingkat produksi. Keluaran yang dihasilkan oleh aplikasi adalah visualisasi *bubblechart.* Visualisasi *bubblechart* berfungsi sebagai tampilan dalam metode *Kohonen* dan *output* berupa anggota dari setiap *cluster*.

4.3.4. Analisa *Kohonen* untuk Kasus *Clustering* Sumur Minyak

Pada Tugas Akhir ini digunakan jaringan syaraf tiruan model *Kohonen*. Agar jaringan syaraf tiruan ini mencapai tingkat keandalan yang diinginkan, maka

dibutuhkan inisialisasi yang mencakup seluruh masukan yang mungkin diterima oleh aplikasi.

Adapun langkah-langkah penyelesaian aplikasi pengelompokkan daerah produksi minyak bumi dengan metode *Kohonen* ini dapat digambarkan melalui *flowchart* pada gambar 4.1:



Gambar 4.1. Flowchart Algoritma *Kohonen* secara keseluruhan

Dari gambar *flowchart* diatas dapat dijelaskan bahwa langkah pertama inisialisasi bobot inputan dengan cara random, laju belajar dan ketetanggan topologis. Kemudian masukkan data produksi sumur minyak, lakukan *min max* normalisasi dan hitung similaritas data tersebut dengan menggunakan *euclidian distance*. Cari neuron pemenang dengan melihat neuron yang memiliki similaritas

minimum. Perbaharui bobot neuron pemenang dan tetangganya serta perbarui juga laju belajar (*learning rate*). Jika iterasi belum mencapai maksimum masukkan kembali data masukkan data produksi minyak dan lakukan langkah selanjutnya hingga mencapai iterasi maksimum.

4.3.4.1. Algoritma *Kohonen*

1. Inisialisasi bobot (Wij) dengan nilai random, tingkat pembelajaran (*learning rate*) , dan fungsi tetangga menggunakan *cubic topology*.
2. Masukkan data produksi sumur minyak dalam bentuk format *.data.
3. Lakukan *Min Max* Normalisasi (rumus 2.1). Min max normalisasi melakukan transformasi linier pada attribut data asli guna menghasilkan range nilai yang sama yaitu antara 1 dan 0.
4. Hitung similaritasnya dengan menggunakan jarak *Euclidian* dan pilih neuron pemenang (rumus 2.2). Jarak Euclidian dipilih kerena umum digunakan dan memiliki tingkat kehandalan yang cukup baik (Yusof, 2006)
5. Update bobot neuron pemenang dan tetangganya (rumus 2.3).
6. *Update* tingkat pembelajaran (rumus 2.5) dan fungsi tetangga (rumus 2.6).
7. Lakukan langkah 2 sampai 5 sampai nilai *epoch* tercapai.

Contoh Persoalan Menggunakan Metode *Kohonen*

Misalkan kita ingin mengelompokkan data-data berikut menjadi 2 kelompok:

X1	X2
0,10	0,10
0,20	0,20
0,30	0,10
0,50	0,30
0,40	0,40
0,20	0,40

Bobot awal yang akan kita gunakan adalah matrik berukuran 2x2 dengan tiap-tiap elemen bernilai 0,5. Tingkat pembelajaran (α = 0,6) dengan setiap kenaikan iterasi akan diset 0,5x(α). Radius bernilai 0. Maksimum iterasi ditetapkan sebesar 10.

Penyelesaian :

1. Inisialisasi:

   α = 0,60

   Pengurangan α = 0,5

   σ = 0

   MaxIterasi = 10

   Bobot awal , W_{ij} :

   0,50 0,50

   0,50 0,50

2. Matrik yang akan dikelompokkan, X :

   0,10 0,10

   0,20 0,20

   0,30 0,10

   0,50 0,30

   0,40 0,40

   0,20 0,40

   Matrik di atas tidak memerlukan normalisasi, karena nilai matrik berada diantara 1 dan 0.

3. Hitung similaritas dengan dengan menggunakan jarak Euclidian dan *update* bobot neuron pemenang dan tetangganya:

   Iterasi ke - 1

   Data ke - 1

   Jarak (*d*) pada :

   Rumus : $d = \sum_{i}^{n} (Wi - Xi)^2$

   Bobot ke-1 = $(0,5 - 0,1)^2 + (0,5 - 0,1)^2 = 0,32$

   Bobot ke-2 = $(0,5 - 0,1)^2 + (0,5 - 0,1)^2 = 0,32$

   Jarak terkecil pada bobot ke-1

   Bobot ke-1 baru :

   Rumus : $W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)]$

   $W_{11} = 0,5 + 0,6\ (0,1 - 0,5) = 0,26$

   $W_{12} = 0,5 + 0,6\ (0,1 - 0,5) = 0,26$

Data ke - 2

Jarak pada :

Rumus : $d = \sum_{i}^{n} (Wi - Xi)^2$

Bobot ke-1 = $(0,26 - 0,2)^2 + (0,26 - 0,2)^2 = 0,0072$

Bobot ke-2 = $(0,5 - 0,2)^2 + (0,5 - 0,2)^2 = 0,1800$

Jarak terkecil pada bobot ke-1

Bobot ke-1 baru :

Rumus : $W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)]$

$W_{11} = 0,26 + 0,6\ (0,2 - 0,26) = 0,224$

$W_{12} = 0,26 + 0,6\ (0,2 - 0,26) = 0,224$

Data ke - 3

Jarak pada :

Rumus : $d = \sum_{i}^{n} (Wi - Xi)^2$

Bobot ke-1 = $(0,224 - 0,3)^2 + (0,224 - 0,1)^2 = 0,0212$

Bobot ke-2 = $(0,5 - 0,3)^2 + (0,5 - 0,1)^2 = 0,2000$

Jarak terkecil pada bobot ke-1

Bobot ke-1 baru :

Rumus : $W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)]$

$W_{11} = 0,224 + 0,6\ (0,3 - 0,224) = 0,2696$

$W_{12} = 0,224 + 0,6\ (0,1 - 0,224) = 0,1496$

Data ke - 4

Jarak pada :

Rumus : $d = \sum_{i}^{n} (Wi - Xi)^2$

Bobot ke-1 = $(0{,}2696 - 0{,}5)^2 + (0{,}1496 - 0{,}3)^2 = 0{,}0757$

Bobot ke-2 = $(0{,}5 - 0{,}5)^2 + (0{,}5 - 0{,}3)^2 = 0{,}0400$

Jarak terkecil pada bobot ke-2

Bobot ke-2 baru :

Rumus : $W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)]$

$W_{21} = 0{,}5 + 0{,}6\ (0{,}5 - 0{,}5) = 0{,}5000$

$W_{22} = 0{,}5 + 0{,}6\ (0{,}3 - 0{,}5) = 0{,}3800$

Data ke - 5

Jarak pada :

Rumus : $d = \sum_{i}^{n} (Wi - Xi)^2$

Bobot ke-1 = $(0{,}2696 - 0{,}4)^2 + (0{,}1496 - 0{,}4)^2 = 0{,}0797$

Bobot ke-2 = $(0{,}5 - 0{,}4)^2 + (0{,}38 - 0{,}4)^2 = 0{,}0104$

Jarak terkecil pada bobot ke-2

Bobot ke-2 baru :

Rumus : $W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)]$

$W_{21} = 0{,}5 + 0{,}6\ (0{,}4 - 0{,}5) = 0{,}4400$

$W_{22} = 0{,}38 + 0{,}6\ (0{,}4 - 0{,}38) = 0{,}3920$

Data ke - 6

Jarak pada :

Rumus : $d = \sum_{i}^{n} (Wi - Xi)^2$

Bobot ke-1 = $(0,2696 - 0,2)^2 + (0,1496 - 0,4)^2 = 0,0675$

Bobot ke-2 = $(0,44 - 0,2)^2 + (0,392 - 0,4)^2 = 0,0577$

Jarak terkecil pada bobot ke-2

Bobot ke-2 baru :

Rumus : $W_{ij}(t+1) = W_{ij}(t) + \alpha(t)h(t) * [X_i(t) - W_{ij}(t)]$

$W_{21} = 0,44 + 0,6\ (0,2 - 0,44) = 0,2960$

$W_{22} = 0,392 + 0,6\ (0,4 - 0,392) = 0,3968$

4. *Update* tingkat pembelajaran dan fungsi tetangga.

   Tingkat pembelajaran : $\alpha = 0,5 * 0,6 = 0,3$

   Fungsi tetangga : 0

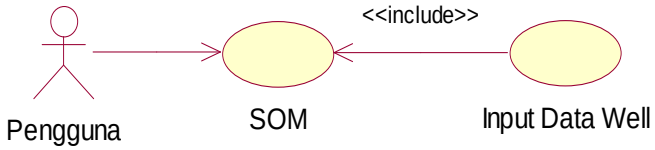
5. Lakukan langkah 2 sampai 4 sampai iterasi maksimum.

4.4. Perancangan Aplikasi

Setelah analisis sistem selesai dilakukan, maka akan dilakukan perancangan terhadap model aplikasi yang akan dibuat. Model perancangan yang akan digunakan dalam aplikasi ini adalah UML meliputi *use case diagram, class diagram, sequence diagram, collaboration diagram, activity diagram, statechart diagram* dan *deployment diagram*.

4.4.1. Use Case Diagram

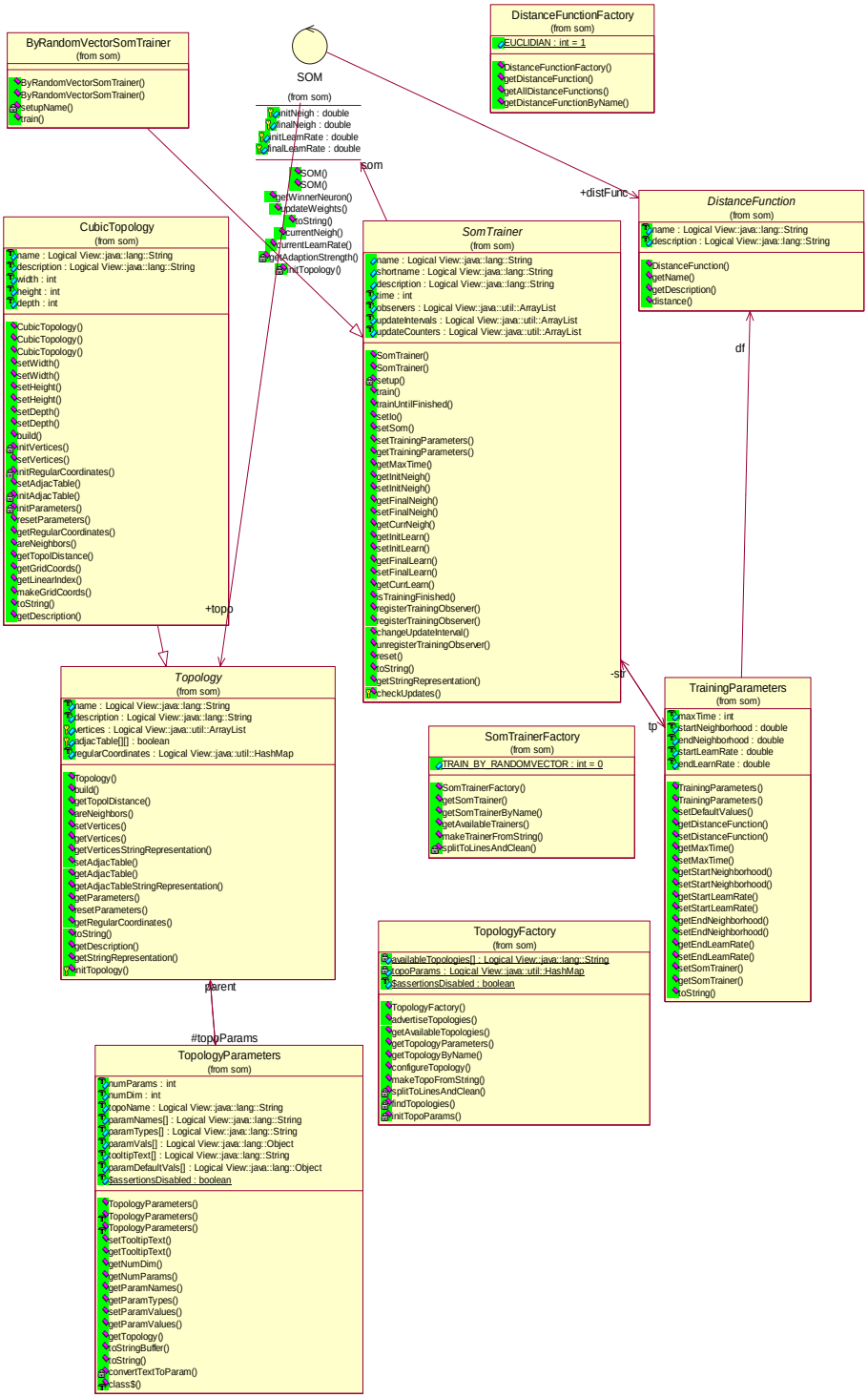
Use Case Diagram menjelaskan mamfaat sistem jika dilihat menurut pandangan orang yang berada di luar sistem (*actor*). Diagram ini menunjukkan fungsionalitas sistem dan bagaimana cara sistem berinterakasi dengan perangkat luar dari sistem atau aplikasi. Pada *Use Case Diagram* ini pengguna aplikasi melakukan proses pengelompokkan dengan cara menginputkan *data well* ke dalam sistem, kemudian *data well* tersebut akan di eksekusi oleh aplikasi SOM.



Gambar 4.2 Model *Use Case Diagram*

4.4.2. Class Diagram

Class Diagram menggambarkan struktur dan deskripsi *class*, *package*, dan objek beserta hubungan satu sama lain, yang digunakan dalam melakukan pembuatan *project* baru, membuka *project*, menghapus *project*, melakukan eksekusi dengan Algoritma *Kohonen (SOM)*, dan memvisualisasikan hasil pengelompokkan ke dalam bentuk *bubblechart.*



Gambar 4.3. *Class Diagram* Algoritma SOM

Tabel 4.1 *Class Diagram* Algoritma SOM

No.	Nama Kelas atau Objek	Deskripsi
1.	SOM	*Class* yang berfungsi melakukan menjalankan algoritma Kohonen (SOM)
2.	*DistanceFunction*	*Class* yang berfungsi menentukan *Distance Euclidian*
3.	*Topology*	Class yang berfungsi menentukan topology yang digunakan
5.	CubicTopology	*Class* untuk topology yang berbentuk kotak
6.	ByRandomVectorSomTrainer	*Class* untuk melatih algoritma SOM dengan menggunakan nilai random
7.	*SomTrainer*	*Class* proses palatihan algoritma SOM
8.	TrainingParameters	*Class* yang menentukan parameter untuk pelatihan
9.	TopologyFactory	*Class* yang menyimpan data topology
10.	DistanceFunctionFactory	*Class* yang menyimpan fungsi *Distance*
11.	SomTrainerFactory	*Class* yang menyimpan data pelatihan

Perancangan *sequence diagram, collaboration diagram, activity diagram, statechart diagram* dan *deployment diagram* terlampir pada lampiran B.

4.5. Lingkungan Perancangan

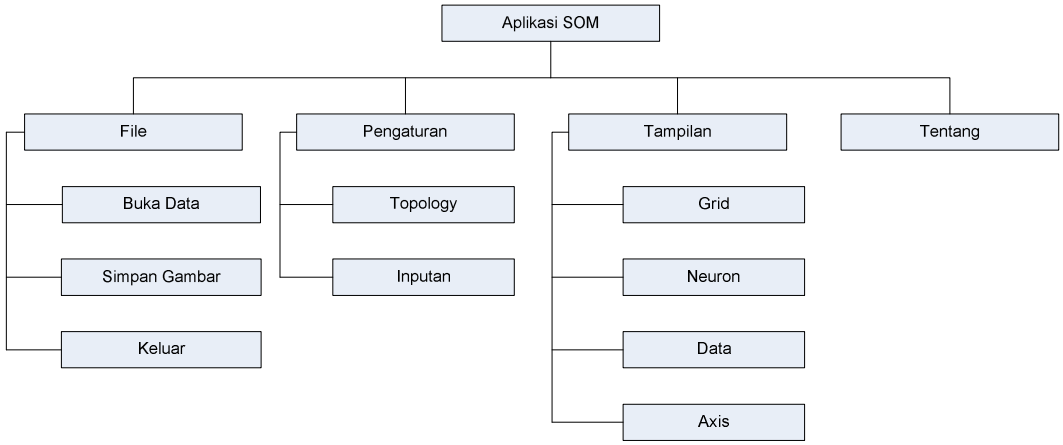
Lingkungan perancangan yang digunakan adalah sebagai berikut:

1. Perangkat Keras, yaitu komputer dengan spesifikasi:
   a. Prosesor : *Intel Core 2 Duo* 1.66 GHz
   b. Memori : 1 GB RAM
   c. Harddisk : 120 GB
2. Perangkat Lunak
   a. Sistem Operasi : *Microsoft Windows XP*
   b. Bahasa Pemograman : Java

4.6. Perancangan Antar Muka

4.6.1. Perancangan Struktur Menu

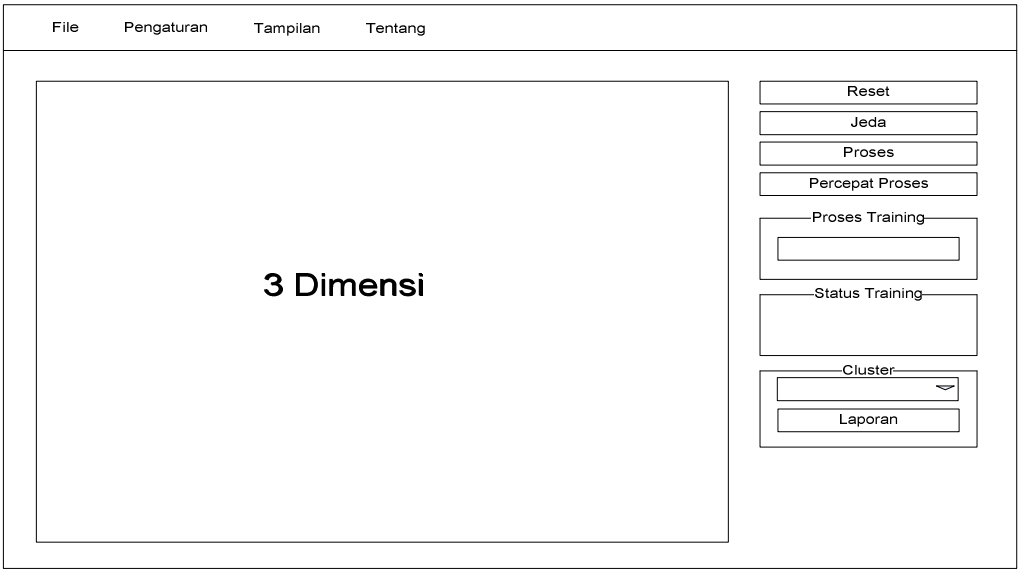
Tujuan perancangan adalah untuk membuat panduan pada tahap implementasi mengenai rancangan dari aplikasi yang akan dibuat. Masalah yang akan diselesaikan adalah pengelompokkan daerah produksi minyak. Perancangan antar muka secara diagram dapat dilihat pada gambar 4.4.



Gambar 4.4. Struktur Menu

4.7. Perancangan Tampilan

Agar aplikasi ramah pengguna, maka perlu dirancang tampilan-tampilan yang mudah dimengerti pengguna, sehingga pengguna mudah menggunakan aplikasi ini. Berikut ini beberapa rancangan tampilan yang sesuai dengan perancangan struktur menu yang dibuat. Untuk spesifikasi perancangan tampilan sistem/antar muka yang lebih rinci dapat dilihat pada lampiran C.



Gambar 4.5. Tampilan Utama

Tabel 4.2. Spesifikasi Objek Tampilan Menu Utama

Nama objek	Jenis	Keterangan
Menu File	MenuBar	Berfungsi ntuk membuka data, menyimpan gambar dan keluar dari aplikasi
Menu Pengaturan	MenuBar	Berfungsi untuk pengaturan topology yang digunakan dan memasukkan inputan.
Menu Tampilan	MenuBar	Berfungsi untuk mengatur tampilan
Menu Tentang	MenuBar	Berfungsi untuk menampilkan tentang pembuat aplikasi.
Menu Reset	Button	Berfungsi untuk me*reset* pelatihan
Menu Jeda	Button	Berfungsi untuk menangguhkan pelatihan
Menu Proses	Button	Berfungsi untuk memulai pelatihan
Menu Percepat Proses	Button	Berfungsi untuk mempercepat pelatihan
Menu Proses Training	TextBox	Berfungsi untuk menampilkan proses yang sedang berjalan
Menu Laporan	Button	Berfungsi untuk menampilkan laporan

BAB V

IMPLEMENTASI DAN PENGUJIAN

5.1 Implementasi Sistem

Implementasi aplikasi pengelompokkan daerah produksi minyak bumi dengan menggunakan jaringan syaraf tiruan metode *Kohonen* ini dibangun dengan bantuan bahasa pemrograman *Java.*

5.1.1 Alasan Pemilihan Perangkat Lunak

Pemilihan perangkat lunak ini didasarkan pertimbangan sebagai berikut:

1. Berorientasi *Object, java* telah menerapkan konsep pemograman berorientasi *object* yang modern dalam implementasinya.
2. *Protable*, program *java* dapat berjalan pada sistem operasi apapun yang miliki *Java Virtual Machine*.
3. Netral secara arsitektur, *java* tidak terikat pada suatu mesin atau sistem operasi tertentu.
4. Terdistribusi, *java* didesain untuk berjalan pada lingkungan yang terdistribusi seperti halnya *internet*.

5.1.2 Batasan Implementasi

Batasan implementasi pada penulisan Tugas Akhir ini adalah aplikasi menampilkan visualisasi 3 dimensi dari pengelompokkan daerah produksi minyak bumi.

5.1.3 Lingkungan Implementasi

Tempat dibangunnya aplikasi sistem pengenalan karakter menggunakan jaringan syaraf tiruan, dan tempat dilakukan pengujian terhadap hasil dari aplikasi, dilakukan dalam lingkungan implementasi yang terdiri atas perangkat keras dan perangkat lunak, yaitu:

Perangkat keras, yaitu komputer dengan spesifikasi

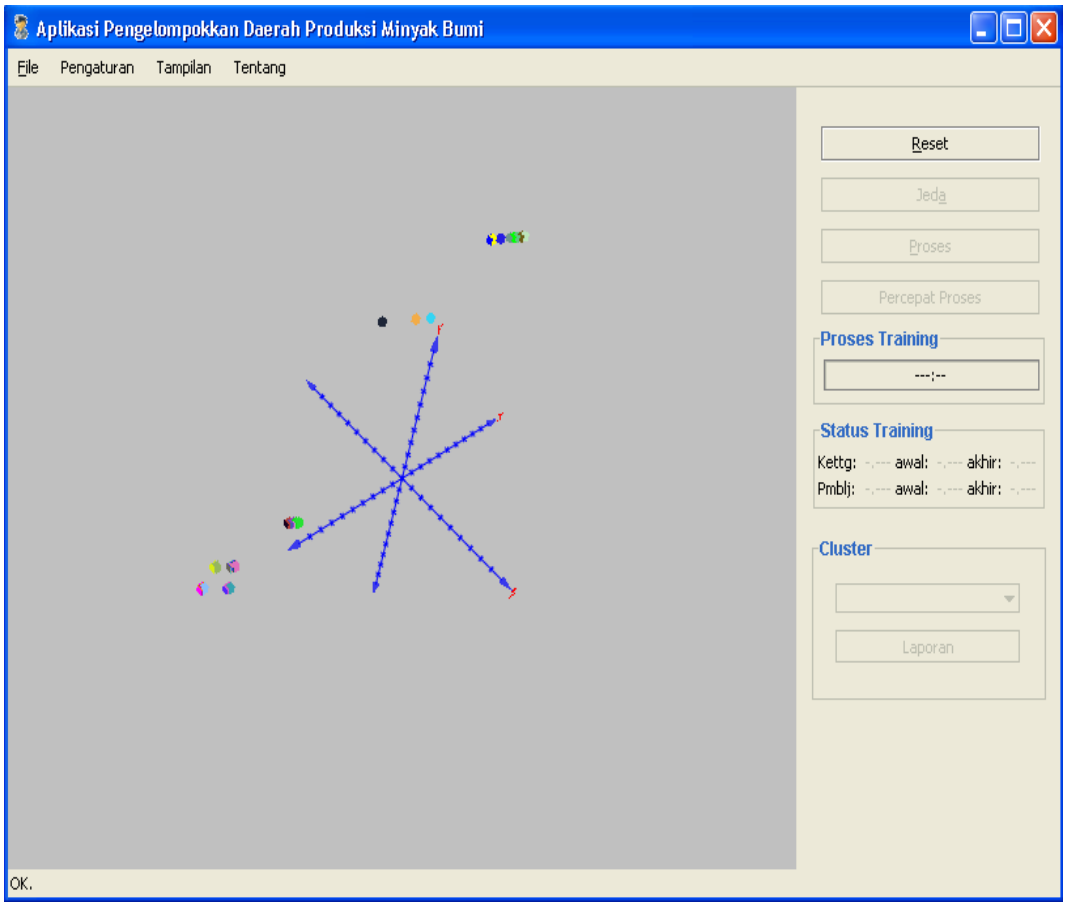
1. Prosesor : *Intel Core* 2 *Duo* 1.66 GHz
2. Memori : 1 GB RAM
3. *Harddisk* : 120 GB

Perangkat Lunak

1. Sistem Operasi : *Microsoft Windows XP Profesional*
2. Bahasa Pemrograman : *Java*

5.1.4 Implementasi Aplikasi Pengelompokkan Daerah Produksi Minyak Bumi

Berikut ini hasil implementasi dari aplikasi pengelompokkan daerah produksi minyak bumi dengan menggunakan metode *Kohonen***.**



Gambar 5.1. Menu Utama

Menu yang terdapat pada Menu Utama, yaitu:

1. Menu File

   Digunakan untuk membuka data, menyimpan gambar dan keluar dari aplikasi.

2. Menu Pengaturan

   Digunakan untuk mengatur *topology* dan mengatur inputan untuk pelatihan.

3. Menu Tampilan

   Digunakan untuk mengatur tampilan output seperti neuron, grid, data dan axis.

4. Menu Tentang

   Menu yang menampilkan perihal tentang pembuat aplikasi.

5. Tombol Riset

   Tombol yang berfungsi me-*reset* proses pelatihan.

6. Tombol Jeda

   Tombol yang berfungsi menangguhkan pelatihan.

7. Tombol Proses

   Tombol yang berfungsi untuk memulai pelatihan.

8. Tombol Percepat Proses

   Tombol yang berfungsi untuk mempercepat pelatihan.

9. Menu Model Tampilan

   Menu yang mengatur model tampilan dari *output*.

10. Menu Proses Training

    Digunakan untuk menampilkan lamanya proses *training*.

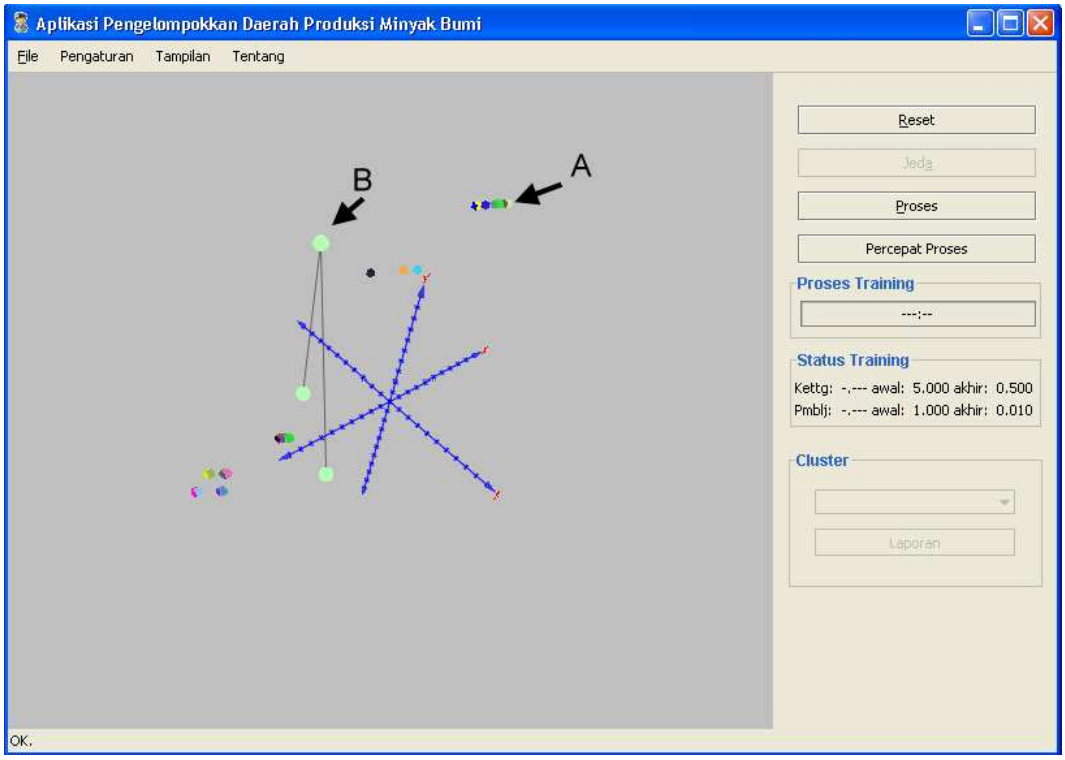
11. Menu Status Training

    Menu yang menampilkan status pelatihan.

12. Menu Cluster

Digunakan untuk menampilkan hasil *cluster* dan laporan.

5.1.4.1 Tampilan Proses Pengelompokkan



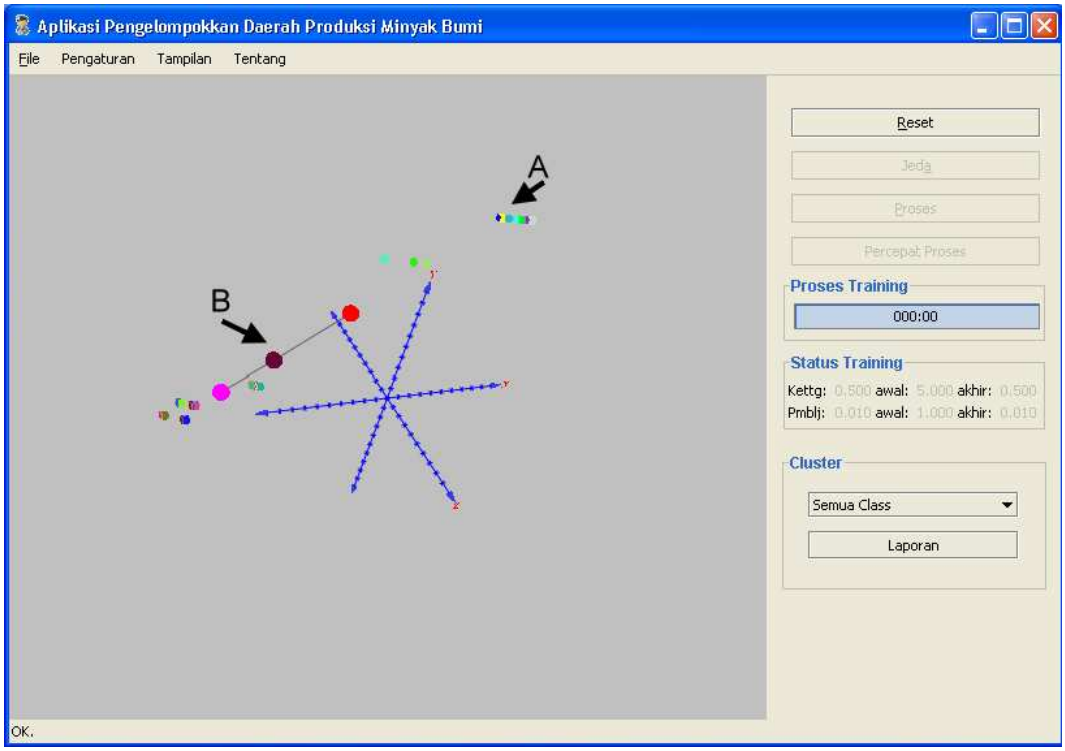
Gambar 5.2 Tampilan Sebelum Pengelompokkan

Keterangan Gambar :

A : Bulatan kecil yang mewakili sumur minyak

B : Bulatan Besar yang mewakili *cluster* (kelompok)

Pada gambar 5.2 bulatan besar yang mewakili *cluster* masih memiliki warna yang sama karena anggota dari setiap *cluster* belum diketahui, sumur minyak masih berkelompok secara acak.



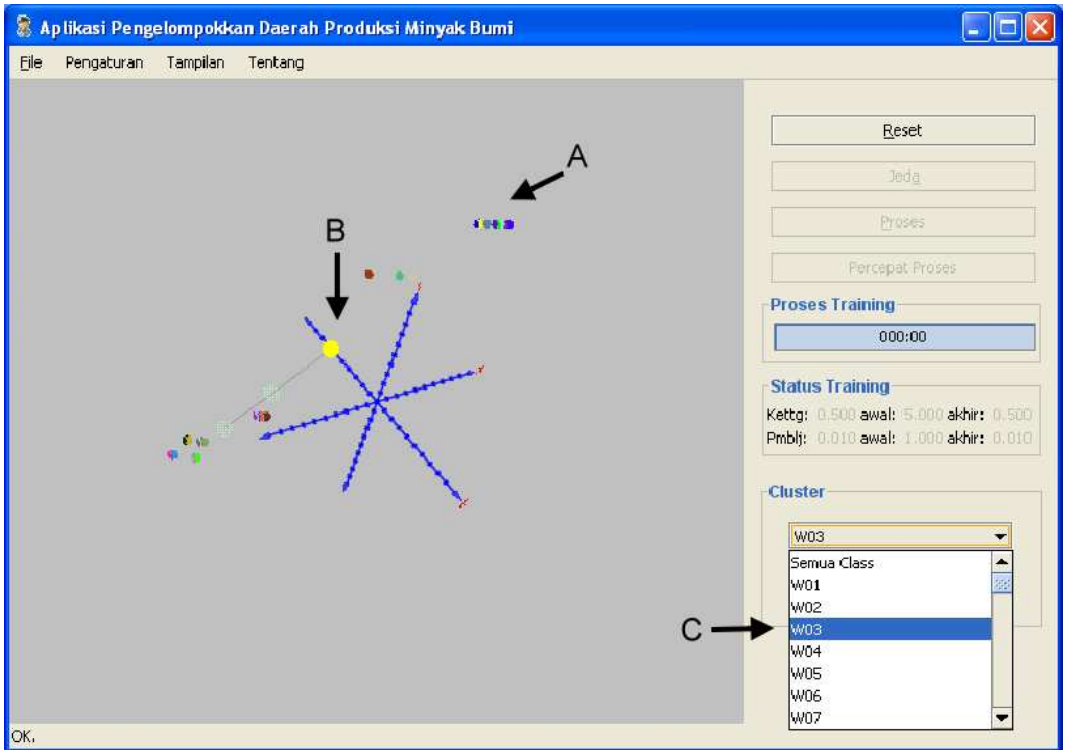
Gambar 5.3 Tampilan Sebelum Pengelompokkan

Keterangan Gambar :

A : Bulatan kecil yang mewakili sumur minyak

B : Bulatan Besar yang mewakili *cluster* (kelompok)

Pada gambar 5.3 bulatan besar yang mewakili *cluster* memiliki warna yang berbeda karena setiap *cluster* sudah mempunyai anggota dimana anggota tersebut merupakan satu atau lebih sumur minyak.



Gambar 5.4 Tampilan Melihat Kelompok Sumur Minyak

Keterangan Gambar :

A : Bulatan kecil yang mewakili sumur minyak

B : Bulatan Besar yang mewakili *cluster* (kelompok)

C : *Combo Box* yang berguna untuk melihat satu atau keseluruhan sumur minyak

Pada gambar 5.4 bulatan besar yang mewakili *cluster* hanya satu yang berwarna karena W3 atau sumur minyak yang dipilih merupakan anggota dari *cluster* tersebut. Implementasi secara rinci dapat dilihat pada lampiran D.

5.2 Pengujian

5.2.1 Pengujian *White Box*

Pengujian *white box* merupakan pengujian yang dilakukan dengan menguji setiap *procedure* yang digunakan dalam sistem. Pengujian ini bertujuan untuk mengetahui apakah *output* dari setiap *procedure* sudah benar atau tidak sehingga dapat dipastikan *output* akhir juga benar. Pengujian *white box* selengkapnya dapat dilihat pada lampiran E.

5.2.2 Pengujian Aplikasi

Pengujian dilakukan dengan tujuan untuk menjamin aplikasi yang dibuat sesuai dengan hasil analisis dan perancangan serta menghasilkan satu kesimpulan. Tahap ini dilakukan untuk mencari segala kemungkinan kesalahan yang timbul dan memeriksa apakah perangkat lunak tersebut telah sesuai dengan yang diharapkan.

Tabel 5.2. Pengujian Aplikasi

No	Iterasi	α awal	α akhir	Radius awal	Radius akhir	% Error	Waktu
1	1000	1	0.001	5	0.5	0	1:46
2	1000	0.001	0.0001	5	0.5	0	1:32
3	1000	0.9	0.02	5	0.5	0	1:18
4	1000	0.1	0.01	5	0.5	0	1:20
5	2000	0.6	0.05	5	0.4	0	1:54
6	1500	0.3	0.01	5	0.2	0	1:32

Tabel 5.2. Hasil Pengelompokkan oleh Aplikasi

Cluster	Sumur Minyak	Minyak	Air	Gas
1	5	44	99	0
	6	45	99	0
	7	46	99	0

Cluster	Sumur Minyak	Minyak	Air	Gas
1	8	47	99	0
	13	79	98	0
	14	76	98	0
	15	77	98	0
	16	78	98	0
	18	46	99	0
	20	80	98	0
	22	47	99	0
	24	76	99	0
	26	46	99	0
	28	75	99	0
	30	47	99	0
	32	75	99	0
	34	46	99	0
	36	76	99	0
	38	46	99	0
	40	77	99	0
	42	46	99	0
	44	74	99	0
	46	59	98	0
	48	75	98	0
	50	56	98	0
2	9	142	96	0
	10	142	96	0
	11	141	96	0
	12	144	96	0
	19	144	96	0
	23	137	96	0
	27	148	96	0
	31	141	96	0
	35	144	96	0
	39	144	96	0
	43	136	96	0
	45	227	86	0
	47	145	96	0
3	1	375	82	0

Cluster	Sumur Minyak	Minyak	Air	Gas
3	3	351	82	0
	4	349	82	0
	17	360	82	0
	21	360	82	0
	25	387	82	0
	29	379	82	0
	33	384	82	0
	37	371	82	0
	41	283	86	0
	49	266	86	0

Pada tabel di atas dapat disimpulkan *cluster* 1 (pertama) nilai minyak berkisar dari 44 sampai 80 dan air dari 98 sampai 99, kemudian pada *cluster* 2 (kedua) nilai minyak berkisar dari 227 sampai 145 dan air dari 86 sampai 96, terakhir *cluster* 3(ketiga) nilai minyak berkisar dari 266 sampai 387 dan air dari 82 sampai 86.

5.2.3 Pengujian *Matlab*

Pengujian dilakukan dengan tujuan untuk menjamin aplikasi yang dibuat sesuai dengan perhitungan matematik. Tahap ini dilakukan untuk mencari segala kemungkinan kesalahan yang timbul dalam perhitungan pada aplikasi. Pengujian secara rinci dijelaskan pada lampiran F.

5.2.4 Lingkungan Pengujian Aplikasi

Pengujian sistem ini dilakukan pada lingkungan perangkat lunak *matlab 5.3* dan perangkat keras sama dengan lingkungan implementasi

5.2.5 Kesimpulan Hasil Pengujian

Berdasarkan hasil pengujian, aplikasi pengelompokkan daerah produksi minyak bumi menggunakan metode *Kohonen* mampu melakukan pengelompokkan dengan optimal, setelah membandingkan hasil pengujian aplikasi dengan pengujian menggunakan *Matlab.* Lamanya proses bergantung pada :

1. Jumlah Iterasi

   Banyaknya jumlah iterasi sangat mempengaruhi lamanya proses pengelompokkan dikarenakan banyaknya terjadi *loop program*.

2. Laju Pembelajaran

   Semakin besar laju pembelajaran atau nilai α(t), semakin cepat bobot koneksi beradaptasi / semakin besar pengaruh vektor input terhadap perubahan bobot koneksi yang terjadi. *Learning rate* ini semakin lama akan semakin mengecil, berkurang seiring berjalannya waktu/iterasi. Semakin dekat *learning rate* α(t) mendekati 0, perubahan bobot akan semakin kecil dan vektor-vektor input dapat dipetakan dengan baik.

3. Radius Ketetanggaan

   Semakin lama nilai dari fungsi ini semakin kecil, berkurang seiring berjalannya waktu / iterasi. Semakin lama pengaruh dari perubahan bobot semakin menyempit, akhirnya hanya *neuron best matching* yang dipengaruhi. Fungsi ini dipengaruhi oleh *Learning rate*.

Aplikasi melakukan pengelompokkan dengan melihat kemiripan dari inputan. Hasil dari pengelompokkan tidak menggambarkan penuh, setengah

penuh dan kering, karena dibutuhkan analisa dari *user*. Pada pengujian dapat disimpulkan *cluster* 1 (pertama) nilai minyak berkisar dari 44 sampai 80 dan air dari 98 sampai 99, kemudian pada *cluster* 2 (kedua) nilai minyak berkisar dari 227 sampai 145 dan air dari 86 sampai 96, terakhir *cluster* 3(ketiga) nilai minyak berkisar dari 266 sampai 387 dan air dari 82 sampai 86.

BAB VI

PENUTUP

1. Kesimpulan

Kesimpulan dari penelitian tugas akhir ini adalah sebagai berikut:

1. Aplikasi pengelompokkan daerah produksi minyak bumi dapat mengelompokkan daerah produksi minyak bumi sesuai dengan cluster yang diinginkan yaitu 3 dan menampilkan laporan dari setiap *cluster* tersebut.
2. Aplikasi pengelompokkan daerah produksi minyak bumi menggunakan algoritma *Kohonen* yang sudah dibangun ini memiliki kekurangan yaitu *user* harus menganalisa hasil *cluster* untuk menetukan kelompok dari *cluster* tersebut yang terdiri dari penuh, kering dan separuh penuh.

2. Saran

Agar sistem ini bermanfaat dan berdaya guna dimasa sekarang dan yang akan datang, maka penulis memberikan saran sebagai berikut:

1. Aplikasi dapat dikembangkan dan diimplementasikan sehingga dapat langsung menganalisa kelompok dari *cluster*, tanpa harus dilakukan analisa kembali oleh *user*.

2. Aplikasi dapat dikembangkan untuk menganalisa lebih dari 3 *cluster* dengan menambahkan parameter lain yang menunjang keakuratan *clusterisasi.*

DAFTAR PUSTAKA

Akprind, "Pengantar untuk Pemrograman Matlab" [*Online*] Available http://elista.akprind.ac.id/upload/files/4544_Modul2.pdf, diakses 7 Mei 2010

Asep, "*Modul Pemrograman Java*" [*Online*] Available www.asephs.web.ugm.ac.id/.../MODUL%20PEMROGRAMAN%20JAVA/.../BAB%20I%20PENDAHULUAN.pdf, diakses 5 Mei 2010

Budhi, Gregorius Satia, Liliana dan Steven Harryanto, *"Cluster Analysis untuk Memprediksi Talenta Pemain Basket Menggunakan Jaringan Saraf Tiruan Self Organizing-Map (SOM) "*, UK Petra, Surabaya, 2006

Daryanto, *"Pengetahuan Dasar Ilmu Komputer "*, Yrama Widya, Bandung, 2003

Hartono, Jogiyanto, *"Analisis dan Disain Sistem Informasi : Pendekatan Terstruktur Teori dan Apikasi Bisnis*", Edisi II, Andi Offset, Yogyakarta, 1999

Haykin, Simon, *"Neural Network A Comprehensive Foundation*", Edisi 2, Halaman 465-501, Pearson Prentice Hall, Singapore, 2005

Hermawan, Arif, *"Jaringan Saraf Tiruan: Teori dan Aplikasi"*, Penerbit Andi, Yogyakarta, 2006

Iriani,Sumini, *"Perancangan dan Implementasi Perangkat Lunak Visualisasi 2D untuk Mengetahui Penyebaran Minyak Pada Suatu Lapangan Produksi Minyak Dengan Metode Clustering"*, UIN SUSKA, Pekanbaru, 2007

ITS Student, "*Jaringan Saraf Tiruan (Neural Networ)*" [*Online*] Available http://Student.eepis-its.edu/~prara/Semester%205/AI/Bab%208%20Jaringan%20Syaraf%20 Tiruan.pdf, diakses 9 Agustus 2009

Jha, Girish Kumar. *"Artificial Neural Networks and Its Applications"*, I.A.R.I, New Delhi, 2007

Kusumadewi, Sri, *"Artificial Intelligence (Teknik dan Aplikasinya)"*, Graha Ilmu, Yogyakarta, 2003

Kusumadewi, Sri, *"Membangun Jaringan Syaraf Tiruan Menggunakan Matlab & Excel Link"*, Graha Ilmu, Yogyakarta, 2004

Madarum, Arum, *"Clustering Specimen Daun Dikotiledon Dengan Menggunakan SOM"*, Institut Pertanian Bogor, Bogor, 2003

Nugroho, Adi, "*Rational Rose untuk Pemodelan Berorientasi Objek"*, Informatika, Bandung, 2004

Nugroho, Eko, *"Pengenalan Komputer"*, Penerbit Andi Offset, 1993

Pressman, Roger S, *"Rekayasa Perangkat Lunak Pendekatan Praktisi (Buku Dua)"*, Penerbit ANDI, Yogyakarta, 1997

Rao, V.B dan Rao, H.V, *"Neural Network and Fuzzy Logic"*, Management Information Source, New York, 1993

Somantri, Maman, *"Pemrograman Berorientasi Objek Menggunakan Java"*, Universitas Diponegoro, Semarang, 2004

Suhendar, A. Gunadi Hariman, "*Visual Modelling Menggunakan UML dan Rational Rose"*, Informatika, Bandung, 2002

Sutopo, Ariesto Hadi, *"Analisa dan Desain Berorientasi Objek"*, J & J Learning, Yogyakarta, 2002

Valpola, Harri, "*Supervised vs Unsupervised Learning*" [*Online*] Available http://www.cis.hut.fi/harri/thesis/valpola_thesis/node34.html, diakses 8 Mei 2010

"_____", *"Eksplorasi Minyak Bumi*" [*Online*] Available http://id.wikipedia.org/wiki/Eksplorasi_minyak_bumi, diakses 5 Mei 2010

"_____", *"Geologi Minyak Bumi*" [*Online*] Available http://id.wikipedia.org/wiki/Geologi_minyak_bumi, diakses 5 Mei 2010

"_____", *"Jaringan Syaraf Tiruan*" [*Online*] Available http://id.wikipedia.org/wiki/Jaringan_saraf_tiruan, diakses 5 Mei 2010

"_____", *"Dari Mana Datangnya Minyak Bumi?*" [*Online*] Available http://wiki.migas-indonesia.net/index.php/Minyak_Bumi, diakses 9 Agustus 2009

Yusof, Norfadzila Binti Mohd, "*Multilevel Learning in Kohonen Som Network for Classsification Problems"*, Universiti Teknologi Malaysia, Malaysia, 2006

Zemke, Stefan, *"Data Mining for Prediction Financial Series Case"*, The Royal Institute Technology, Sweden, 2003